COMBINATORIAL APPROACH TO ABV-PACKETS FOR GL_N

CONNOR DAVID RIDDLESDEN
Bachelor of Science, Coventry University, 2020

A thesis submitted
in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in

MATHEMATICS

Department of Mathematics and Computer Science
University of Lethbridge
LETHBRIDGE, ALBERTA, CANADA

COMBINATORIAL APPROACH TO ABV-PACKETS FOR $\mathbf{GL_N}$

CONNOR DAVID RIDDLESDEN

Date of Defence: August 12, 2022

Dr. A. Fiori Thesis Supervisor	Associate Professor	Ph.D.
Dr. A. Akbary Thesis Examination Committee Member	Professor	Ph.D.
Dr. H. Kharaghani Thesis Examination Committee Member	Professor	Ph.D.
Dr. J. Sheriff Chair, Thesis Examination Committee	Assistant Professor	Ph.D.

Dedication

To my Mum, who has always believed in me, and my Dad, who still cannot pronounce the word "combinatorial".

Abstract

There exists a significant conjecture in the local Langlands correspondence that A-packets are ABV-packets. For the case $G = GL_n$, the conjecture reduces to ABV-packets for orbits of Arthur type being singletons, which is a specialisation of the wider conjecture known as the Open-Orbit conjecture. We can reduce the complexity of this problem by considering the combinatorial geometry of these objects using multisegments, since there exists a natural relationship between this description and the structure of ABV-packets. The first part of this thesis investigates interpretations of the Zelevinskii Involution. We then use combinatorial approaches involving endoscopic decompositions and numerical invariants to study the partial ordering in the Open-Orbit conjecture, which will lead to the proof that ABV- packets for orbits of Arthur type in GL_n are singletons. Finally, we use a numerical-based argument to conjecture families of ABV-packets for which the partial ordering relation is not satisfied for.

Acknowledgements

First and foremost, I would like to thank Andrew Fiori for his continued support and guidance as a supervisor and mentor throughout this journey. I am extremely grateful to Andrew for allowing me to jump down the various rabbit holes in the project and the space which he has afforded me to grow my own research style. I am also grateful for the freedom and advocation that Andrew has provided me to explore various opportunities away from my thesis. Andrew you are exceptional individual and apart from helping me achieve to my academic goals, you have taught me a tremendous amount about myself which I am eternal grateful for. It has truly been an honour to work with you.

Furthermore, I wish to thank all members of the Department of Mathematics and Computer Science for creating the best possible environment to complete my studies and enduring my incessant pacing around corridors throughout my time in Lethbridge. A special thank you goes to my fellow Graduate students and colleagues (especially Joel, Solaleh, Eli, and David) for countless hours of interesting conversations, tea breaks, and helping make my graduate student experience brilliant. I would like to thank Amir Akbary, Hadi Kharaghani and John Sheriff for being members of my supervisory committee, for reading my thesis and giving their thorough feedback on the thesis. I would also like to thank Amir Akbary and Sean Fitzpatrick, who I learned a lot from whilst TAing their courses over the last two years. I am also grateful to Bobby Miraftab, who provided a side-project that was a welcome distraction from this thesis and a really enjoyable collaboration. I would be remiss in not mentioning the Voganish group, who provided an incredibly complex but enjoyable and interesting project for this thesis.

I would like to express my deepest gratitude to Matthew Lamaudière, not only for his essential help with navigating the world of academia but also for standing by me as a friend, motivating me and supporting me during difficult times. My gratitude also goes to Igor Morozov, who has always showed great belief in my ability and continued to be a guiding light. Thanks should also go to Ha Tran, Amy Feaver and Paul Griffiths, who helped start my journey in research and pushed me to come to Lethbridge.

I would like to extend my sincere thanks to the Stephen Family for an incredible end to my time in Lethbridge. Denise and Bill, words cannot express how grateful I am to you both for opening up your home to me and taking me in as one of your family. Alex, Olivia and Sarah, thank you for being so exceptional and inspiring me to be the best version of myself. I hope each one of you has a great time in your new adventures!

Finally, a huge thank you must go out to my family and friends back in the United Kingdom, who have continued to be of great support whilst I follow my dreams despite the distance and time difference. My parents, Andrea and Simon, for their love and inspiration throughout my life. Apologies for the copious amount of hours you both had to listen to mathematical ramblings and practice attempts for presentations.

Table of Contents

List of Tables

List of Figures

Chapter 1 : Introduction

The Langlands program is an extremely important set of conjectures in number theory and geometry. Inside the Langlands program lies the local Langlands correspondence for which there exists a significant conjecture: A-packets are ABV-packets. Both A-packets and ABV-packets will be finite sets of irreducible representations of GL_n when restricting to $G = GL_n$, which we will do in this thesis. This significant conjecture was first presented by Cunningham et al. [4]. Further for $G = GL_n$, the conjecture reduces to ABV-packets for orbits of Arthur type being singletons, which is a specialisation of the wider conjecture known as the Open-Orbit conjecture. The primary motivation for this thesis has been to study the Open-Orbit conjecture for different families of ABV-packets and prove the conjecture that ABV-packets for orbits of Arthur type in GL_n are singletons. The study of these conjectures requires one to also study an involution on the set of irreducible representations of GL_n, commonly known as the Zelevinskii involution. In this study we will be required to look at the geometry of moduli spaces of Langlands parameters which was first discussed by Zelevinskii. For a wider study of the geometry of moduli spaces of Langlands parameters including those groups outside of $G = GL_n$ see the work of Cunningham et al. [5].

The approach taken to the problem in this thesis avoids the need to understand highly technical objects about which the conjectures are actually made. Instead we will focus on the combinatorial geometry of these technical objects since one can more easily deduce properties about them. One method for studying the combinatorial geometry is through the use of multisegments, which have become an effective combinatorial depiction for objects in representation theory. Multisegments simply consist of a collection of sequences of integers and can be used to describe conjugacy classes of orbits of Langlands parameters. One method of realising the Zelevinskii involution on multisegments is through the Mœglin-Waldspurger algorithm. The Zelevinskii involution also has an interesting combinatorial application for the so-called Schützenberger involution in the theory of Young tableaux. Knight and Zelevinskii in [9] use a multisegment depiction to prove that the Schützenberger involution and Mœglin-Waldspurger algorithm emit equivalent results. Therefore the desire to study the Mœglin-Waldspurger algorithm is also highly motivated by its additional applications to the Young tableaux, which has further applications to both Schubert Calculus and Schur functions.

Chapter 2 begins with the background material concerning how the project fits in to the local Langlands program, describes the geometry of moduli spaces of Langlands parameters, and their simplification to a multisegment description. Following this it introduces the Zelevinskii involution, which is interpreted as an involution on multisegments, along with the Open-Orbit conjecture, and a partial ordering of multisegments.

Chapter 3 studies various methods for implementing the Zelevinskii involution. The first method which is presented in Section 3.1 is the Mœglin-Waldspurger algorithm which is implemented on the multisegment description. Knight and Zelevinskii in [9] show that there exists a method of using the maximum flow through a network to implement the Zelevinskii involution. In Section 3.2, we define an analogous network to Knight and Zelevinskii's, and relate this to the Mœglin-Waldspurger algorithm. Furthermore, we discusses how the Mœglin-Waldspurger algorithm can be implemented on the network, which leads to a greater understanding of the algorithm.

Our study of the Open-Orbit conjecture for ABV-packets is contained in Chapter 4. The key is to study the interaction of a partial ordering and an involution which, perhaps surprisingly, does not always respect this ordering. We employ two combinatorial techniques in this process:

1. Endoscopic Decomposition: We partition a multisegments into sub-multisegments and then run the algorithm on each of the individual partitions.
2. Numerical Invariants: This categorises the multisegments using various numerical properties and creates various combinatorial optimisation problems based on these properties.

These techniques will then be used to categorise various families of ABV-packets that either satisfy or fail to satisfy a relation which is understood to imply a packet is a singleton and will be introduced in Chapter 2. This leads to the most significant result of the thesis: *ABV-packets for orbits of Arthur type in GL_n are singletons*, which is shown using a numerical argument in Section 4.2.

Finally, Chapter 5 concludes the thesis with a brief overview of areas which require further research, and questions that still remain.

Chapter 2 : Background and Motivation

The background material for this thesis is contained between Section 2.1 and Section 2.4 inclusively. Following on from the background, there is a main focus on establishing how the relations discussed in the Open-Orbit conjecture in Section 2.4 can be reflected by both a rank and multisegment description in Section 2.5.

The complete breakdown of each section in the chapter is:

2.1 We discuss the necessary background for how the thesis fits into the local Langlands correspondence.

2.2 We fix the case for $G = GL_n$ and study how the objects can be represented by "rank triangles".

2.3 We introduce the notion of a multisegment and give an explicit algorithm to swap between the rank triangle and multisegment descriptions.

2.4 We present the notion of the conormal bundles and discover how its introduction gives rise to both the Zelevinskii involution and the Open-Orbit conjecture.

2.5 We look into how relations from the Open-Orbit conjecture can be interpreted in terms of a partial ordering on the rank triangles and multisegments.

2.1 Background

The *Langlands program* is a collection of significant conjectures which are described by Edward Frenkel as 'a grand unified theory of mathematics' [8]. This collection of conjectures were first proposed by Canadian mathematician Robert Langlands in a series of papers between 1967 and 1970, hence the eponymous name for the program, and describe a number of relationships between many seemingly unrelated areas of mathematics from number theory, to harmonic analysis, on to geometry and even further to quantum physics [10, 11]. One significant application of the Langlands program was in the solution to *Fermat's Last Theorem* by Wiles and Taylor in 1994 [18].

One specific area of the Langlands program, and the main focus of this research, is the local Langlands correspondence/conjectures. To present the local Langlands correspondence we must set up the notations and introduce a couple of key terms.

Let G be a reductive algebraic group over a p-adic field F, then Langlands derived a new complex group, the *Langlands dual group*, which is denoted by $\hat{G}$. Some important examples of Langlands dual groups are as follows:

Table 2.1: Examples of Langlands Dual Groups

G	GL_n	Sp_{2n}	SO_{2n+1}	SO_{2n}
$\hat{G}$	$GL_n(\mathbb{C})$	$SO_{2n+1}(\mathbb{C})$	$Sp_{2n}(\mathbb{C})$	$SO_{2n}(\mathbb{C})$

Building off this construction one can define an *L-group* to be denoted ${}^LG$ and is such that ${}^LG = W_F \ltimes \hat{G}$[1], where W_F is the dense subgroup of the Galois group and denotes the Weil-Deligne group.

Definition 2.1.1 ([14, Definition 3.2.1.])**.** A *Langlands parameter (L-parameter)* is a continuous homomorphism

$$\phi : W_F \times SL_2(\mathbb{C}) \to {}^LG,$$

that satisfies the properties:

1. If $p : {}^LG \to W_F$ and $q : W_F \times SL_2(\mathbb{C}) \to W_F$ are the natural projections, then $p \circ \phi = q$, i.e., the following diagram commutes

$$\begin{tikzcd} W_F \times SL_2(\mathbb{C}) \arrow[r, "\phi"] \arrow[dr, "q"'] & {}^LG \arrow[d, "p"] \\ & W_F \end{tikzcd}$$

2. $\phi(W_F)$ consists of semisimple elements of ${}^LG$.

3. The restriction of ϕ to $SL_2(\mathbb{C})$ is an algebraic representation.

[1] In this thesis we will only study the case in which $\hat{G}$ is split, thus the semidirect product $W_F \ltimes \hat{G}$ can simply be considered as the direct product $W_F \times \hat{G}$.

Thus we find that the group $\hat{G}$ acts on the set of L-parameters by conjugation. So let us denote the set of equivalence classes of L-parameters as $\Phi(G/F)$.

We can now present the local Langlands correspondence.

> The local Langlands correspondence postulates that there exists a surjective finite to one mapping Ω from the equivalence classes of irreducible admissible representations of G/F to the equivalence classes of L-Parameters $\phi : W_F \times SL_2(\mathbb{C}) \to \hat{G}$. [10, 11]

Note we will thus define the *L-packet of* ϕ to be the inverse image of Φ under this map Ω.

Naturally we only want to consider the conjugacy classes of $\hat{G}$ acting on the set of Langlands parameters, so it is natural to study the stabiliser of the action $H = \hat{G}_\phi$. Both $\hat{G}$ and H are topological groups, thus we can consider $A_\phi = H/H_0$, where H_0 is the connected component of the identity e of H. Note a topological space decomposes into its connected components and by construction A_ϕ will be a finite group.

The *enhanced Langlands parameter* is (ϕ, p) where p is an irreducible representation of A_ϕ. With this we may give a stronger form of the local Langlands correspondence: there exists a bijection between irreducible admissible representations of G/F up to isomorphism and enhanced Langlands parameters (ϕ, p) up to isomorphism. Hence an equivalent formation for the *L-packet for* ϕ is therefore given by those representations associated to each pair (ϕ, p) for the fixed ϕ.

Definition 2.1.2 ([14, Section 3.2.3.])**.** The *infinitesimal parameter* for G is a continuous homomorphism $\lambda : W_F \to {}^LG$ such that λ is a section of ${}^LG \to W_F$ and the image of λ consists of semisimple elements in ${}^LG$.

Now let $\mathbb{F}_q$ be the *residue field* of F where q denotes the cardinality of this residue field and let I_F denote the *inertia subgroup* of F. Then there exists an exact sequence

$$1 \to I_F \to W_F \to \mathrm{Gal}\left(\overline{\mathbb{F}_q}/\mathbb{F}_q\right) \to 1.$$

For any $w \in W_F$, let $d_w \in SL_2(\mathbb{C})$ be such that

$$d_w = \begin{pmatrix} |w|^{\frac{1}{2}} & 0 \\ 0 & |w|^{-\frac{1}{2}} \end{pmatrix},$$

where $|\cdot|: W_F \to \mathbb{R}$ is a fixed norm homomorphism, trivial on I_F and sending a topological generator of $\mathrm{Gal}(\overline{\mathbb{F}_q}/\mathbb{F}_q)$, called Frob, to q .

Given any Langlands parameter $\phi : W_F \times SL_2(\mathbb{C}) \to {}^LG$, let us define the *infinitesimal parameter* λ_ϕ *of* ϕ by

$$\lambda_\phi : W_F \to {}^LG,$$
$$w \mapsto \phi(w, d_w),$$

for all $w \in W_F$. This gives a finite surjective mapping from Langlands parameters ϕ to infinitesimal parameters of λ up to isomorphism.

For the unramified case the inertia group I_F will be contained in the kernel whereas in the ramified case we must consider both the Frobenius and the inertia group I_F. We now will restrict to the unramified case, so without loss of generality we may take $W_F = \langle \mathrm{Frob} \rangle$ then $\lambda = \lambda_\phi(\mathrm{Frob})$. For $G = GL_n$, the unramified infinitesimal parameter $\lambda : W_F \to GL_n(\mathbb{C})$ is determined by the image of the Frobenius element. Note the element $\lambda(\mathrm{Frob})$ is semisimple so we can simply assume that it lies in the subgroup of diagonal matrices

$$\lambda(\mathrm{Frob}) = \mathrm{diag}(q^{a_1}, \ldots, q^{a_n}),$$

where $a_1, \ldots, a_n$ are complex numbers.

From here forward, we will simply just consider the unramified case and fix $\lambda = \lambda(\mathrm{Frob})$. It turns out it is natural to study Langlands parameters, and their associated L-packets, by grouping them by infinitesimal parameters, since each Langlands parameter has a unique infinitesimal parameter. Let us now define this set of Langlands parameters to be $\Lambda = \{\phi \mid \lambda_\phi = \lambda\}$. If we consider the map $SL_2(\mathbb{C}) \to \hat{G}$ then this is a map of Lie groups. Let $\mathfrak{sl}_2$ and $\hat{\mathfrak{g}}$ be the respective Lie algebras of $SL_2(\mathbb{C})$ and $\hat{G}$ then the map $SL_2(\mathbb{C}) \to \hat{G}$ is defined by the mapping of their Lie algebras $\mathfrak{sl}_2 \to \hat{\mathfrak{g}}$.

Every Langlands parameter $\phi \in \Lambda$ has the same infinitesimal parameter. Let us construct the complex variety V_λ such that

$$V_\lambda = q\text{-eigenspace of } \lambda \text{ acting on } \hat{\mathfrak{g}}.$$

We can now define a surjective map from Λ to the slightly simpler space V_λ:

$$\Lambda \to V_\lambda : \phi \mapsto x_\phi := d_\varphi \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix},$$

where $\varphi := \phi^\circ \mid_{SL_2(\mathbb{C})}: SL_2(\mathbb{C}) \to \hat{G}$, thus we can consider V_λ as the moduli space of Langlands parameters [4, Proposition 4.2.2]. We will also denote the centraliser in $\hat{G}$ of λ as H_λ.

Remark 2.1.3. This centraliser H_λ will act on the complex variety V_λ.

Before we define a new type of packet, *ABV-packets*, we must first introduce the notion of $\mathrm{Per}_{H_\lambda}(V_\lambda)$ to denote H_λ *equivariant perverse sheaves* on V_λ [2]. Note for every simple perverse sheaf in $\mathrm{Per}_{H_\lambda}(V_\lambda)$, we can define it in terms of an intersection cohomology sheaf $\mathcal{IC}(C_\phi, \mathcal{L}_p)$, where C_ϕ is the orbit of ϕ and $\mathcal{L}_p$ is a local system on C_ϕ associated to the irreducible representation p of A_ϕ, the equivariant fundamental group. There is a natural bijection between isomorphism classes of simple perverse sheaves $\mathcal{IC}(C_\phi, \mathcal{L}_p)$ (with fixed C_ϕ) and representations of A_ϕ. Consequently, there exists a bijection between the $\mathcal{IC}(C_\phi, \mathcal{L}_p)$ on V_λ and $\Pi_\lambda(G/F)$. The local Langlands correspondence can be interpreted as saying that we have a bijection between the sets of equivalence classes of

$$\text{Simple Perverse Sheaves} \longleftrightarrow \begin{matrix}\text{Orbits of L-parameters} \\ \text{with their} \\ \text{Associated Local System}\end{matrix} \longleftrightarrow \text{Enhanced L-parameters} \longleftrightarrow \text{Representations},$$

$$\{\mathcal{IC}(C_\phi, \mathcal{L}_p)\} \longleftrightarrow \{(C_\phi, \mathcal{L}_p)\} \longleftrightarrow \{(\phi, p)\} \longleftrightarrow \Pi_\lambda(G/F).$$

To partially justify the introduction of these complex objects into the interpretation of the local Langlands correspondence, we remark that the Fourier transform on perverse sheaves agrees with the Aubert involution on admissible representations. This is known for $G = GL_n$ but is conjectured more generally. This intrinsic relationship between the two involutions defined on either side of this bijection is one strong reason to believe the introduction of perverse sheaves into the theory is natural. Let us denote the characteristic cycle of a sheaf to be CC.

[2] In this thesis we will not need to explicitly describe perverse sheaves, since complete descriptions are available throughout the surrounding literature on the project and the concept is not relevant for the overall goals of the thesis.

Definition 2.1.4 ([1])**.** For any H_λ-orbit C in V_λ we may associate an *ABV-packet* for C to consist of the collection of simple perverse sheaves for which $T^*_C(V)$ (the conormal bundle **Definition 2.4.1**) is in the support of $CC(\mathcal{IC}(C_\phi, \mathcal{L}_p))$, which denotes the characteristic cycle of the perverse sheaf $\mathcal{IC}(C_\phi, \mathcal{L}_p)$.

Each A-parameter ψ, defined below, will determine elements in the conormal bundle to the orbit C_{ϕ_ψ}. Thus the introduction of the complex object of simple perverse sheaves instead allows us to study the H_λ-orbits in V_λ and the characteristic cycles of their conormal bundle.

There also exists a natural partial ordering on H_λ-orbits in which we say that $C \leq D$ if $C \subset \overline{D}$. There exists a key property for the characteristic cycles of simple perverse sheaves.

Proposition 2.1.5. *If $[C] \in CC(\mathcal{IC}(D, \mathcal{L}_p))$ then $C \leq D$.*

We will now consider an augmentation of L-parameters, which Arthur introduced [2].

Definition 2.1.6 ([14, Definition 3.4.1.])**.** Let G be a reductive group over F, and ${}^LG$ be an L-group for G. Then we define an *Arthur parameter* (or *A-parameter*) for G to be a homomorphism:

$$\psi : \langle \text{Frob} \rangle \times SL_2(\mathbb{C}) \times SL_2(\mathbb{C}) \longrightarrow {}^LG,$$
$$(w, x, y) \longmapsto \psi^0(w, x, y) \rtimes w,$$

such that:

1. The restriction of $\psi\,|_{W_F \times SL_2}$ is a Langlands parameter for G.

2. The restriction ψ^0 to the last copy of SL_2 is a morphism of algebraic groups.

3. $\psi^0\,|_{W_F}$ has bounded image in the complex topology on $\hat{G}$.

Note the collection of all A-parameters for G is denoted $\Psi(G)$, and there exists an injective map from $\Psi(G)$ to $\Phi(G/F)$ given by

$$\psi \mapsto \phi_\psi(w, x) = \psi(w, x, d_w)^3.$$

[3] The fact that this map is injective is not immediately obvious since it follows from the Jacobson-Morozov Theorem [14]. It will not generally be true that the map also holds the property of being surjective.

Let $\Pi(G/F)$ be the set of equivalence classes of irreducible admissible representations of G/F. Then for each Langland parameter $\phi \in \Phi$ there exists a previously defined L-packet $\Pi_\phi \subset \Pi(G/F)$. In [2] Arthur introduced *A-packets*, which are simply expansions of L-packets [2]. Each A-packet is associated with an A-parameter ψ and is an expansion of the L-packet Π_{ϕ_ψ}. Note for the case in which $G = GL_n$ each A-packet is equal to the associated L-packet, and each A-packet is a singleton.

Furthermore, there exists a significant conjecture which explains a relationship between A-packets and ABV-packets (See [5, 17]).

Conjecture 2.1.7. A-packets are ABV-packets.

For the specific case $G = GL_n$ because A-packets are known to be singletons, the conjecture reduces to ABV-packets for orbits of Arthur type being singletons. It is known that there are ABV-packets that are not singletons (See [5, Section 1.3 - Main Results]). Therefore the study of ABV-packets for orbits of Arthur type is of the upmost importance, and there will be a key focus on examining this family of ABV-packets using various combinatorial approaches in this thesis.

2.2 Specialising to the case of $\mathbf{GL_n}$

In this research, we will simply fix $G = GL_n$. We note that for $G = GL_n$, the A-packets are all singletons and equal to their associated L-packet. It is also known that for the GL_n case the Fourier transform on perverse sheaves agrees with the Aubert involution on admissible representations [7, Corollary 7.3].

In GL_n the objects V_λ and H_λ have very concrete descriptions, in particular they are the direct products of varieties of the form:

$$V_\lambda = \oplus_k \operatorname{Hom}(E_k, E_{k+1}) \quad \text{and} \quad H_\lambda = \oplus_k GL(E_k),$$

where E_k denotes an eigenspace of $\lambda(\mathrm{Frob})$ which is associated to the eigenvalue λ_k. Note the eigenspaces E_k and E_{k-1} have an implied relationship between their associated eigenvalues

$$\lambda_k = q\lambda_{k-1}.$$

Let $f_{i,j}$ denote a map between the eigenspaces $E_i \to E_j$. We will refer to the elements of V_λ as quiver representations and denote them by

$$\underline{f} = (f_{k,k+1}) \in \oplus_k \operatorname{Hom}(E_k, E_{k+1}).$$

Then there exists maps for each $i \leq j$ such that

$$f_{i,j} = f_{j-1,j} \circ \cdots \circ f_{i,i+1}.$$

Each map $f_{i,j}$ has an associated rank $r_{i,j}$, which naturally construct rank triangles as follows

$$\begin{array}{ccccccccc}
r_{1,1} & & r_{2,2} & & r_{3,3} & & r_{4,4} & & \cdots \\
& r_{1,2} & & r_{2,3} & & r_{3,4} & & \ddots & \\
& & r_{1,3} & & r_{2,4} & & \ddots & & . \\
& & & r_{1,4} & & \ddots & & & \\
& & & & \ddots & & & &
\end{array}$$

Remark 2.2.1. Assuming that V_λ and H_λ are defined as above, then their orbits can be classified by the ranks of each map $f_{i,j}$, which will be shown in **Proposition 2.3.7**.

Proposition 2.2.2 ([4, Section 1.5])**.** *The rank triangles of the maps $f_{i,j}$ have three key properties:*

1. $r_{i,j} \leq r_{i,j-1}$,

2. $r_{i,j} \leq r_{i+1,j}$, *and*

3. $r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$, *where* $l < i, k \leq j$.

Proof. Using the notions used in the proof of **Proposition 2.3.7**. Then the proofs of the three key properties are as follows:

1. Since $j-1 < j$ then $\ker(f_{i,j-1}) \subset \ker(f_{i,j})$, and by the rank-nullity theorem $r_{i,j} \leq r_{i,j-1}$.

2. Since $i < i+1$ then $\operatorname{im}(f_{i,j}) \subset \operatorname{im}(f_{i+1,j})$, and hence $r_{i,j} \leq r_{i+1,j}$ directly follows.

3. The relations between the ranks can be expressed by $r_{l,k} - r_{l,j} = \dim\big(\mathrm{im}(f_{l,k}) \cap \ker(f_{k,j})\big)$ and $r_{i,k} - r_{i,j} = \dim\big(\mathrm{im}(f_{i,k}) \cap \ker(f_{k,j})\big)$. Since $l < i$ then $\mathrm{im}(f_{l,k}) \subset \mathrm{im}(f_{i,k})$. Therefore $r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$. □

2.3 Multisegments

We will now present a purely combinatorial interpretation of these quiver representations, which represent elements of V_λ.

Definition 2.3.1. Let us define a *segment* to be a non-empty set of consecutive integers

$$\Delta = (b, b+1, \dots, e-1, e).$$

Then a *multisegment* will be a collection of segments

$$\alpha = \{\Delta_1, \Delta_2, \dots, \Delta_r\},$$

where each segment is indexed by i, for $1 \leq i \leq r$, to differentiate between possible duplicates of segments.

We will see that there is a bijection between rank triangles satisfying **Proposition 2.2.2** and multisegments.

Algorithm : Multisegment Construction Using the rank triangles we can inductively construct multisegments by:

1. In the rank triangle identify the $r_{i,j}$ which is the lowest and leftmost nonzero entry.
2. Add to your multisegment α the segment $\Delta = (i, \dots, j)$.
3. Decrease the rank $r_{k,l}$ by one for each value of k,l such that $i \leq k \leq l \leq j$.
4. Repeat until all $r_{i,j}$ in the triangle are equal to zero.

The following Lemma plays an important role in the proof that the multisegment construction algorithm inductively constructs a multisegment.

Lemma 2.3.2 ([4, Section 1.5])**.** *The multisegment construction algorithm is admissible with the rank triangles, i.e., following each iteration the following properties of rank triangles remain satisfied:*

1. $r_{i,j} \leq r_{i,j-1}$,
2. $r_{i,j} \leq r_{i+1,j}$, *and*
3. $r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$, *where* $l < i, k \leq j$.

Proof. Given an arbitrary rank triangle then using the multisegment construction algorithm let us construct an *iterative triangle* in which $r_{I,J}$ the lowest and leftmost nonzero entry (shown below in red) from the rank triangle.

$$
\begin{array}{ccccccccccccccc}
 & \ddots & & \color{red}{\ddots} & & \color{red}{\ddots} & & \color{red}{\vdots} & & \color{red}{\iddots} & & \color{red}{\iddots} & & \iddots & \\
\cdots & & r_{I-1,J-4} & & \color{red}{r_{I,J-3}} & & \color{red}{r_{I+1,J-2}} & & \color{red}{r_{I+2,J-1}} & & \color{red}{r_{I+3,J}} & & r_{I+4,J+1} & & \cdots \\
 & \iddots & & r_{I-1,J-3} & & \color{red}{r_{I,J-2}} & & \color{red}{r_{I+1,J-1}} & & \color{red}{r_{I+2,J}} & & r_{I+3,J+1} & & \ddots & \\
 & & \iddots & & r_{I-1,J-2} & & \color{red}{r_{I,J-1}} & & \color{red}{r_{I+1,J}} & & r_{I+2,J+1} & & \ddots & & \\
 & & & \iddots & & r_{I-1,J-1} & & \color{red}{r_{I,J}} & & r_{I+1,J+1} & & \ddots & & & \\
 & & & & \iddots & & r_{I-1,J} & & r_{I,J+1} & & \ddots & & & & \\
 & & & & & \iddots & & r_{I-1,J+1} & & \ddots & & & & & \\
 & & & & & & \iddots & & \ddots & & & & & & \\
 & & & & & & & \vdots & & & & & & &
\end{array}
$$

The multisegment construction algorithm then states that each value inside this iterative triangle will be decreased by one.

Firstly to prove 1) and 2) remain satisfied, we can consider a smaller triangle

$$
\begin{array}{ccc}
r_{i,j-1} & & r_{i+1,j} \\
 & r_{i,j} &
\end{array},
$$

and the different possibilities for which it lies in our rank triangle.

Case 1:

The smaller triangle is completely outside the iterative triangle, so $r_{i,j-1}, r_{i+1,j}, r_{i,j}$ all lie outside the iterative triangle. Thus each rank remains unchanged so $r_{i,j} \leq r_{i,j-1}$ and $r_{i,j} \leq r_{i+1,j}$ will still be satisfied.

Case 2:

The smaller triangle is completely inside the iterative triangle, so $r_{i,j-1}, r_{i+1,j}, r_{i,j}$ all lie inside the iterative triangle. Thus each rank will be decreased by 1. Then since $r_{i,j} \leq r_{i,j-1}$ and $r_{i,j} \leq r_{i+1,j}$ were satisfied in the original rank triangle then decreasing each value by 1 will mean $(r_{i,j} - 1) \leq (r_{i,j-1} - 1)$ and $(r_{i,j} - 1) \leq (r_{i+1,j} - 1)$ are satisfied following the iteration.

Case 3:

The smaller triangle overlaps the left diagonal of the iterative triangle, so $r_{i,j-1}, r_{i+1,j}, r_{i,j}$ can be rewritten as $r_{I-1,j-1}, r_{I,j}, r_{I-1,j}$ where only the rank $r_{I,j}$ lies inside the iterative triangle. Thus $r_{I,j}$ will be decreased by 1. Now since $r_{I-1,j-1}$ and $r_{I-1,j}$ lie outside the iterative triangle then they remain unchanged, so $r_{I-1,j} \leq r_{I-1,j-1}$ remains satisfied. Recall that in the original triangle

$$r_{I-1,k} - r_{I-1,j} \leq r_{I,k} - r_{I,j}$$

must be satisfied. Now let $j = J$. Then

$$r_{I-1,k} = r_{I-1,k} - r_{I-1,J} \leq r_{I,k} - r_{I,J},$$

since by construction $r_{I-1,J} = 0$.

Similarly, by construction we also know that $r_{I,J} \geq 1$, so

$$r_{I-1,k} \leq r_{I,k} - r_{I,J} \leq r_{I,k} - 1.$$

Therefore we have proved that $r_{I-1,j} \leq r_{I,j} - 1$ is satisfied.

Case 4:

The smaller triangle overlaps the right diagonal of the iterative triangle, so $r_{i,j-1}, r_{i+1,j}, r_{i,j}$ can be rewritten as $r_{i,J}, r_{i+1,J+1}, r_{i,J+1}$ where only the rank $r_{i,J}$ lies inside the iterative triangle. Thus $r_{i,J}$ will be decreased by 1. Now since $r_{i+1,J+1}$ and $r_{i,J+1}$ lie outside the iterative triangle then they remain unchanged, so $r_{i,J+1} \leq r_{i+1,J+1}$ remains satisfied. Recall that in the original triangle

$$r_{l,J} - r_{l,J+1} \leq r_{i,J} - r_{i,J+1}$$

must be satisfied. Now let $l = I$,

$$r_{I,J} = r_{I,J} - r_{I,J+1} \leq r_{i,J} - r_{i,J+1},$$

since by construction $r_{I,J}$ is the lowest nonzero entry so $r_{I,J+1} = 0$. Similarly, by construction we also know that $r_{I,J} \geq 1$, so

$$r_{i,J+1} \leq r_{i,J} - r_{I,J} \leq r_{i,J} - 1.$$

Therefore we have proved that $r_{i,J+1} \leq r_{i,J} - 1$ is satisfied.

Thus since $1)$ and $2)$ remain satisfied for all possible positions of the smaller triangle, then $1)$ and $2)$ are admissible following the algorithm.

Once again to prove $3)$ remains satisfied, we can consider a number of different cases, however this time we will consider the different permutations in which a box with corners $r_{l,j}, r_{l,k}, r_{i,j}, r_{i,k}$ can lie in our rank triangle.

Case 1:

The box is completely outside the iterative triangle, so $r_{l,j}, r_{l,k}, r_{i,j}, r_{i,k}$ all lie outside the iterative triangle. Thus each rank remains unchanged so

$$r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$$

will still be satisfied.

<u>Case 2:</u>

The box is completely inside the iterative triangle, so $r_{l,j}, r_{l,k}, r_{i,j}, r_{i,k}$ all lie inside the iterative triangle. Thus each rank will be decreased by 1. Then since

$$r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$$

was satisfied in the original rank triangle, then decreasing each value by 1 will mean

$$(r_{l,k} - 1) - (r_{l,j} - 1) = r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j} = (r_{i,k} - 1) - (r_{i,j} - 1)$$

will still be satisfied following the iteration.

<u>Case 3:</u>

The ranks $r_{i,k}, r_{i,j}$ are contained in the iterative triangle, and the rank $r_{l,k}, r_{l,j}$ lies outside the iterative triangle. Thus $r_{i,k}, r_{i,j}$ will both be decreased by 1. Then since

$$r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$$

was satisfied in the original rank triangle, and

$$(r_{i,k} - 1) - (r_{i,j} - 1) = r_{i,k} - r_{i,j}.$$

Then we find that

$$r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j} = (r_{i,k} - 1) - (r_{i,j} - 1)$$

is also true. Thus we have proved that

$$r_{l,k} - r_{l,j} \leq (r_{i,k} - 1) - (r_{i,j} - 1)$$

will still be satisfied following the iteration.

Case 4:

Similarly, the ranks $r_{l,k}, r_{i,k}$ are contained in the iterative triangle, and the rank $r_{l,j}, r_{i,j}$ lies outside the iterative triangle. Thus $r_{l,k}, r_{i,k}$ will both be decreased by 1. Then since

$$r_{l,k} - r_{l,j} \leq r_{i,k} - r_{i,j}$$

was satisfied in the original rank triangle. Then we find that

$$(r_{l,k} - 1) - r_{l,j} = r_{l,k} - 1 - r_{l,j} \leq r_{i,k} - 1 - r_{i,j} = (r_{i,k} - 1) - r_{i,j}$$

is also true. Thus we have proved that

$$(r_{l,k} - 1) - r_{l,j} \leq (r_{i,k} - 1) - r_{i,j}$$

will still be satisfied following the iteration.

Case 5:

Only one corner of the box is contained in the iterative triangle, so only the rank $r_{i,k}$ lies inside the iterative triangle.. Thus $r_{i,k}$ will be decreased by 1 and the rest remain unchanged. We now strive to prove that

$$r_{l,k} - r_{l,j} \leq (r_{i,k} - 1) - r_{i,j}.$$

By construction $r_{l,j} = 0$, so instead we need prove

$$r_{l,k} \leq (r_{i,k} - 1) - r_{i,j}$$

will still be satisfied following the iteration. To prove this we will need to consider a number of subcases:

Case 5a:
The case in which $r_{i,k} = r_{I,J}$, thus $r_{l,J} = 0$ and $r_{I,j} = 0$. Therefore $r_{l,k} = r_{l,J} = 0$ and $(r_{I,J} - 1) - r_{I,j} = r_{I,J} - 1 \geq 0$, since $r_{I,J} > 0$ by the multisegment construction algorithm. Thus

$$r_{l,k} = r_{l,J} = 0 \leq r_{I,J} - 1 = (r_{I,J} - 1) - r_{I,j} = (r_{i,k} - 1) - r_{i,j},$$

so $r_{l,k} \leq (r_{i,k} - 1) - r_{i,j}$ remains satisfied.

Case 5b:
The case in which $r_{i,k} = r_{I,k}$ for $k < J$, thus $r_{I,j} = 0$. Therefore what remains is to prove

$$r_{l,k} \leq (r_{I,k} - 1) - r_{I,j} = r_{I,k} - 1.$$

Now by rule 2), we know that $r_{l,k} \leq \cdots \leq r_{I-1,k} \leq r_{I,k} - 1$ is satisfied.

Case 5c:
The case in which $r_{i,k} = r_{i,J}$ for $i > I$, thus $r_{l,J} = 0$. Therefore what remains is to prove

$$r_{l,k} = r_{l,J} = 0 \leq (r_{i,J} - 1) - r_{i,j}.$$

Now by rule 1), we know that $r_{i,j} \leq \cdots \leq r_{i,J+1} \leq r_{i,J} - 1$, so $0 \leq (r_{i,J} - 1) - r_{i,j}$ is satisfied.

Case 5c:
Finally, if $i > I$ and $k < J$ then we have two relations from the original rank triangles

$$\begin{aligned} r_{I,k} - r_{I,J} &\geq r_{l,k} - r_{l,J} = r_{l,k}, && \text{(E1)} \\ r_{i,J} - r_{i,j} &\geq r_{I,J} - r_{I,j} = r_{I,J}, \\ r_{i,J} - r_{I,J} &\geq r_{i,j}, && \text{(E2)} \end{aligned}$$

Now taking the sum of E1 and E2 we find

$$(r_{i,J} + r_{I,k}) - 2r_{I,J} \geq r_{l,k} + r_{i,j}.$$

We also have the relation $r_{i,k} - r_{i,J} \geq r_{I,k} - r_{I,J}$ from the original triangle which gives $r_{i,k} + r_{I,J} \geq r_{I,k} + r_{i,J}$. Thus

$$r_{i,k} - r_{I,J} = (r_{i,k} + r_{I,J}) - 2r_{I,J} \geq (r_{i,J} + r_{I,k}) - 2r_{I,J} \geq r_{l,k} + r_{i,j}.$$

Thus $r_{i,k} - 1 \geq r_{l,k} + r_{i,j}$ since $r_{I,J} \geq 1$ by the multisegment construction algorithm. Therefore we have found $(r_{i,k} - 1) - r_{i,j} \geq r_{l,k}$ is satisfied.

Therefore we have proved that

$$r_{l,k} \leq (r_{i,k} - 1) - r_{i,j}$$

will still be satisfied following the iteration for all possible cases. $\square$

Definition 2.3.3. Let α be a multisegment, then let us define the *multiplicity of a segment* $(i, \ldots, j)$, denoted by $m_{i,j}$ to be the number of times in which the segment $(i, \ldots, j)$ appears in α.

Given that the multisegment construction algorithm inductively forms multisegments. Then the following proposition provides an incredibly quick alternative for computing the algorithm.

Proposition 2.3.4. *The multiplicity of a segment $(i,\dots,j)$ in the multisegment is given by*

$$m_{i,j} = r_{i,j} - r_{i-1,j} - r_{i,j+1} + r_{i-1,j+1}.$$

Note that we assume that the rank is equal to 0 *if it is not defined inside the rank triangle.*

Proof. Let us now consider the following formation,

$$\begin{array}{ccc} & r_{i,j} & \\ r_{i-1,j} & & r_{i,j+1} \\ & r_{i-1,j+1} & \end{array}.$$

The multiplicity of the segment $(i,i+1,\dots,j-1,j)$ is given by the number of times in which the multisegment construction algorithm is carried out for which $r_{i,j}$ is the lowest and leftmost nonzero entry of the rank triangle, so the point of the iterative triangle. Thus we consider the number of different cases in which $r_{i,j}$ is decreased by 1. Since the algorithm only adds the segment $(i,i+1,\dots,j-1,j)$ to the multisegment when $r_{i,j} > 0$ and $r_{i-1,j} = r_{i,j+1} = r_{i-1,j+1} = 0$. Thus

$$m_{i,j} = r_{i,j} - (\text{The cases in which } r_{i,j} \text{ is not at the point of the iterative triangle}).$$

Now let us consider these different cases:

1. The case in which $r_{i-1,j}$, $r_{i-1,j+1}$, $r_{i,j}$, $r_{i,j+1}$ are all contained inside the iterative triangle, then there are $r_{i-1,j+1}$ possibilities for this.

2. The case in which only $r_{i-1,j}$ and $r_{i,j}$, are all contained inside the iterative triangle, then there are $(r_{i-1,j} - r_{i-1,j+1})$ possibilities for this.

3. The case in which only $r_{i,j}$ and $r_{i,j+1}$, are all contained inside the iterative triangle, then there are $(r_{i,j+1} - r_{i-1,j+1})$ possibilities for this.

Therefore we have found that

$$\begin{aligned} m_{i,j} =& r_{i,j} - \left[r_{i-1,j+1} + (r_{i-1,j} - r_{i-1,j+1}) + (r_{i,j+1} - r_{i-1,j+1})\right], \\ =& r_{i,j} - r_{i-1,j} - r_{i,j+1} + r_{i-1,j+1}. \end{aligned}$$

□

Similarly to the ranks, we can naturally construct a triangle for the multiplicity of the segments in the multisegment using **Proposition 2.3.4** as follows

$$\begin{array}{ccccccccc}
m_{1,1} & & m_{2,2} & & m_{3,3} & & m_{4,4} & & \cdots \\
 & m_{1,2} & & m_{2,3} & & m_{3,4} & & \ddots & \\
 & & m_{1,3} & & m_{2,4} & & \ddots & & . \\
 & & & m_{1,4} & & \ddots & & & \\
 & & & & \ddots & & & &
\end{array}$$

Unsurprisingly, there also exists a converse algorithm that forms the rank triangle from the associated multisegment and basis.

Algorithm : Rank Triangle Construction Given a multisegment then we can inductively construct a rank triangle by:

1. Construct the initial rank triangle by letting $r_{i,j} := 0$ for all $\{i, j\}$ such that $m \leq i < j \leq n$, where m and n respectively denote the minimum and maximum values appearing in any of the segments.

2. For each segment $\Delta = (b, \ldots, e)$ in multisegment α increase $r_{i,j}$ by one for each value $b \leq i < j \leq e$.

Note this algorithm can be simply implemented by using the formula

$$r_{i,j} = \sum_{n \leq i \leq j \leq k} m_{n,k}.$$

Proposition 2.3.5. *The* Multisegment Construction *and* Rank Triangle Construction *algorithms are inverses.*

Proof. Firstly, we can use the formula

$$r_{i,j} = \sum_{n \leq i \leq j \leq k} m_{n,k}$$

to find the rank $r_{i,j}$ from the multisegment description. Now recall that the multisegment construction algorithm can be implemented using

$$m_{i,j} = r_{i,j} - r_{i-1,j} - r_{i,j+1} + r_{i-1,j+1}.$$

If we now substitute this into the formula then we should find that the right-hand side of the equation will hence be equal to $r_{i,j}$. Note that there must exist a minimum and maximum value for n,k which we will denote a and b respectively.

$$\begin{aligned}
r_{i,j} &= \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} m_{n,k}, = \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} (r_{n,k} - r_{n-1,k} - r_{n,k+1} + r_{n-1,k+1}),\\
&= \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n,k} - \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n-1,k} - \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n,k+1} + \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n-1,k+1},\\
&= \sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n,k} - \sum_{\substack{a-1\leq n'\leq i-1\\ j\leq k\leq b}} r_{n',k} - \sum_{\substack{a\leq n\leq i\\ j+1\leq k'\leq b+1}} r_{n,k'} + \sum_{\substack{a-1\leq n'\leq i-1\\ j+1\leq k'\leq b+1}} r_{n',k'},\\
&= \left(\sum_{\substack{a\leq n\leq i\\ j\leq k\leq b}} r_{n,k} - \sum_{\substack{a-1\leq n'\leq i-1\\ j\leq k\leq b}} r_{n',k}\right) + \left(- \sum_{\substack{a\leq n\leq i\\ j+1\leq k'\leq b+1}} r_{n,k'} + \sum_{\substack{a-1\leq n'\leq i-1\\ j+1\leq k'\leq b+1}} r_{n',k'}\right),\\
&= \left(\sum_{j\leq k\leq b} r_{i,k} - \sum_{j\leq k\leq b} r_{a-1,k}\right) + \left(- \sum_{j+1\leq k'\leq b+1} r_{i,k'} + \sum_{j+1\leq k'\leq b+1} r_{a-1,k'}\right),
\end{aligned}$$

Note that $a-1$ is less than the minimum value so we can assume all rank $r_{a-1,l} = 0$ for all l. This will also be true for any ranks for $b+1$.

$$= \sum_{j\leq k\leq b} r_{i,k} - \sum_{j+1\leq k'\leq b+1} r_{i,k'} = r_{i,j} - r_{i,b+1} = r_{i,j}.$$

The rank and multisegment triangles are vector spaces of the same dimension. We have defined linear maps between them through the *Multisegment Construction* and *Rank Triangle Construction* algorithms. We have then proved above that the composition of two linear maps form an identity map. Therefore we have found that the two algorithms will be inverses. □

Similarly, we can also construct a canonical quiver representation from a multisegment.

Algorithm : Quiver Representation Construction Given a multisegment α then let us first define a vector space W_k for each integer k appearing in a segment in α, whose basis is indexed by $\Delta \in \alpha$ and $k \in \Delta$, so the basis will be

$$\{\vec{e}_{\Delta,k} \mid \Delta \in \alpha, k \in \Delta\}.$$

Note that the dimensions of each space is thus determined by the total number of appearances of each k in segments of α. A multisegment will then determine the equivalence classes of the quiver representation $\underline{g}$ for a series of maps which take $g_{i,j} : W_i \to W_j$ according to the rule

$$g_{i,j}(\vec{e}_{\Delta,i}) = \begin{cases} \vec{e}_{\Delta,j}, & j \in \Delta; \\ 0, & \text{otherwise.} \end{cases}$$

Remark 2.3.6. The rank triangle of the associated quiver representation is the rank triangle constructed by the rank triangle construction.

Proposition 2.3.7. *Assuming that V_λ and H_λ are defined as in Section 2.2, then their orbits can be classified by the ranks of each map*

$$f_{i,j} = f_{j-1,j} \circ \cdots \circ f_{i,i+1}.$$

Proof. By Krull-Remak-Schmidt's ([6, Theorem 1.7.4]) and Gabriel's ([6, Theorem 4.4.13]) Theorems, every orbit has a representative of the form produced by the quiver representation construction from a uniquely determined multisegment. By **Proposition 2.3.5** and **Remark 2.3.6** there is a bijection between such quiver representations and rank triangles. $\square$

2.4 Zelevinskii Involution

We are now in a position to be able to present the conormal bundle for orbits in V and consequently the Zelevinskii involution. Following this we will then present the Mœglin-Waldspurger algorithm in **Section 3.1** which gives an algorithm for computing the Zelevinskii involution using multisegments.

$$\begin{tikzcd}
 & & & & & & \mathbf{V} \\
E_1 \arrow[r, bend left, "f_1"] & E_2 \arrow[l, bend left, "g_1"] \arrow[r, bend left, "f_2"] & E_3 \arrow[l, bend left, "g_2"] \arrow[r, bend left, "f_3"] & E_4 \arrow[l, bend left, "g_3"] \arrow[r, bend left, dashed] & E_{r-1} \arrow[l, bend left, dashed] \arrow[r, bend left, "f_{r-1}"] & E_r \arrow[l, bend left, "g_{r-1}"] & \\
 & & & & & & \mathbf{V}^*
\end{tikzcd}$$

Figure 2.1: The maps between vector spaces

Naturally there also exists a *dual space* associated with the vector space V over the field F which is denoted V^*.

Using the duality on

$$\mathrm{Hom}(E_i, E_{i+1})^* \sim \mathrm{Hom}(E_{i+1}, E_i),$$

where the duality of the homomorphisms is defined by

$$\langle f, g \rangle = \mathrm{tr}(f \circ g).$$

We can then express

$$V^* = \oplus Hom(E_{j+1}, E_j).$$

The dual space consists of all linear maps $\theta : V \to F$ and will also form a vector space. If we now denote the respective H-orbits of V and V^* as C and D then their respective orbits will be in terms of the f_i's and g_i's given in the **Figure 2.1**.

Definition 2.4.1 ([5, Proposition 6.3.])**.** The conormal bundle of C for V is denoted $T^*_C(V)$ and given by

$$T^*_C(V) = \{(f, g) \in V \times V^* \mid f \in C, f \circ g = g \circ f\}.$$

The notion of the *conormal bundle* is most commonly defined using differential geometry in terms of manifolds. Therefore this Lie theoretic description is very much unusual. From above the duality of the homomorphisms is defined by

$$\langle f, g \rangle = \mathrm{tr}(f \circ g).$$

Recall that both V and V^* are part of the Lie algebra $\mathfrak{g}$ and $\mathrm{tr}(f \circ g)$ is (up to scaling) the restriction of the *Killing Form* on $\mathfrak{g}$.

There naturally exists a projection map from the conormal bundle $T^*_C(V)$ to V^*,

$$\begin{aligned} \rho : \ T^*_C(V) &\longrightarrow V^*, \\ (f, g) &\longmapsto g. \end{aligned}$$

Proposition 2.4.2 ([5, Lemma 6.5.]). *Given the projection map* $\rho : T_C^*(V) \longrightarrow V^*$, *then*

$$\overline{\rho(T_C^*(V))} = \overline{D}$$

is well defined.

This involution is often referred to as the *Zelevinskii involution* [19].

Proof. Firstly observe $T_C^*(V)$ has a natural group action of $\prod GL(E_i)$ which determines D uniquely. The closure $\overline{\rho(T_C^*(V))}$ will be a union of orbits. Note there will only be finitely many orbits, and $\overline{\rho(T_C^*(V))}$ is closed, thus it is clearly the union of at most finitely many closures of orbits. We also know that $\overline{\rho(T_C^*(V))}$ is irreducible, that is, not the union of two disjoint closed subsets. If it were, then, so too is $\rho(T_C^*(V))$, and by continuity of the projection map $T_C^*(V)$ would also be reducible. But $T_C^*(V)$ is irreducible because it is a vector bundle over $C = H/Stab_C$, which is irreducible because $H = \prod GL(E_i)$ is irreducible. □

Note from this point on D will no longer be used to denote the orbits in V^* and will instead be used to denote a new H_λ orbit.

Proposition 2.4.3 ([16, Corollary 2]). *The map on the set of orbits given by* $C \mapsto C^*$ *is a bijection from* H*-orbits in* V *to* H*-orbits in* V^*.

Further there exists a relationship between characteristic cycles of the intersection cohomology sheaf and their Fourier transform denoted by FT:

Proposition 2.4.4 ([4, Proposition 3.2.1]). *If* $\mathcal{IC}(C_\phi, \mathcal{L}_p)$ *is a simple perverse sheaf then*

$$CC(FT(\mathcal{IC}(C_\phi, \mathcal{L}_p))) = CC(\mathcal{IC}(C_\phi, \mathcal{L}_p))^*.$$

In the context of GL_n *this gives that*

$$FT(\mathcal{IC}(C, \mathcal{L}_1)) = \mathcal{IC}(C^*, \mathcal{L}_1),$$

where the 1 *in* $\mathcal{L}_1$ *denotes the trivial representation of the trivial group.*

Hence there exists a corollary to **Proposition 2.1.5** for GL_n [7, Corollary 7.3]:

Corollary 2.4.5. *For the case of GL_n, if $[C] \in CC(\mathcal{IC}(D, \mathcal{L}_1))$ then $C^* \leq D^*$.*

Remark 2.4.6. Therefore if $[C] \in CC(\mathcal{IC}(D, \mathcal{L}_1))$ then we find $C \leq D$ and $C^* \leq D^*$, which may seem counterintuitive since one may expect that following the involution the inequality will reverse to instead be $D^* \leq C^*$. Thus it is quite reasonably to expect that in many cases

$$CC(\mathcal{IC}(D, \mathcal{L}_1)) = [D].$$

In Chapter 4, we will show a number of examples of orbits for which the inequalities $C \leq D$ and $C^* \leq D^*$ are both satisfied.

This leads us to present a significant conjecture for ABV-packets in GL_n:

Conjecture 2.4.7 (Open Orbit)**.** For the case of GL_n, suppose $T^*_C(V)$ has an open H-orbit. If $C \in CC(\mathcal{IC}(D, \mathcal{L}_p))$ then

$$C = D,$$

hence the ABV-packet for C is a singleton.

Remark 2.4.8. There are examples of orbits, for which $T^*_C(V)$ does not have an open orbit, for which the conclusion of the conjecture does not hold (See [5]).

The study of this conjecture is not a trivial problem, so the remainder of this thesis will be devoted to setting up the notation for which we can then study conjecture for various families of Langlands parameters.

2.5 Partial Ordering Relation

We will now define a partial ordering relation for rank triangles based upon the containment conditions for H-orbits defined in Section 2.4:

Proposition 2.5.1 ([4, Section 1.5])**.** *$C \subset \overline{D}$ if and only if $c_{i,j} \leq d_{i,j}$ for all i, j, where $c_{i,j}$ and $d_{i,j}$ are the associated ranks $r_{i,j}$ for C and $\overline{D}$ respectively.*

There also exists a partial ordering relation for multisegments which we can study following the introduction of the following action between any two segments of the multisegment.

Proposition 2.5.2. *Let* Δ_1 *and* Δ_2 *be any two segments in an arbitrary multisegment* α*, then we can construct a new multisegment* β *by replacing each* Δ_1 *and* Δ_2 *in* α *with respectively*

$$\begin{cases} \Delta_1 \cap \Delta_2 \textit{ and } \Delta_1 \cup \Delta_2, & \textit{if } \Delta_1 \cap \Delta_2 \neq \emptyset \textit{ and } \Delta_1 \neq \Delta_2; \\ \Delta_1 \cup \Delta_2, & \textit{else if } \Delta_1 \cup \Delta_2 \textit{ is a segment}; \\ \Delta_1 \textit{ and } \Delta_2, & \textit{otherwise}. \end{cases}$$

then we have that $\alpha \leq \beta$.

Note we denote the first action as the *union intersection* and the second as *conjunction*. The third action will simply leave the multisegment unchanged.

We will now introduce a new and extremely important relation between pairs of segments in a multisegment. This relation will be used to continue our study of actions on multisegments and further to an algorithm which can be used to compute the associated dual multisegment.

Definition 2.5.3 ([13]). Given any two segments $\Delta_1 = (b_1, \ldots, e_1)$ and $\Delta_2 = (b_2, \ldots, e_2)$, then we say that Δ_1 *precedes* Δ_2 if $b_1 < b_2$, $e_1 < e_2$, and $b_2 \leq e_1 + 1$.

There also exists an implicit relationship between the actions on pairs of segments and the preceding relation.

Proposition 2.5.4. *If we take a single action on* α *to generate* β*, then* $\alpha \neq \beta$ *if and only if the two segments which we take the action on in* α *have a preceding relation.*

Proof. Firstly, let $\Delta_1 = (b_1, \ldots, e_1)$ and $\Delta_2 = (b_2, \ldots, e_2)$ be the two segments in α for which we take an action on to form β. Without loss of generality, we can assume that $e_1 \geq e_2$.

Let us assume $\alpha \neq \beta$, then by construction there will be $n_\alpha - 2$ segments in α and β which will be identical since they remain fixed by the single action. Note a single action on α will replace Δ_1 and Δ_2 with either one or two segments. If the action only creates one segment, then it uses the conjunction action and $\alpha \neq \beta$ since $n_\beta < n_\alpha$. To use conjunction action the two segments must be such that $e_1 > e_2$, $b_1 > b_2$ and $e_2 = b_1 - 1$, thus Δ_2 will precede Δ_1. Alternatively, if the action creates two segments then the union intersection action must have been used, since $\alpha \neq \beta$. By definition $\Delta_1 \cap \Delta_2 \neq \emptyset$ and $e_1 \geq e_2$, so $e_2 > b_1 - 1$. Also note that neither Δ_1 or Δ_2 should be completely contained in the other, otherwise, their union intersection would simply form Δ_1 and Δ_2. So $e_1 \neq e_2$

hence $e_2 < e_1$ following the assumption, and $b_2 < b_1$, because we already know $e_2 > b_1 - 1$ and if $b_2 \geq b_1$ then Δ_2 would be completely contained in Δ_1. Therefore if the action creates two segments then $e_2 < e_1$, $b_2 < b_1$ and $e_2 > b_1 - 1$, thus Δ_2 will precede Δ_1. Consequently, both cases imply that the two segments which we take the action on in α have a preceding relation.

Conversely, let us assume the two segments Δ_1 and Δ_2 which we take the action on in α have a preceding relation, so Δ_2 will precede Δ_1. Therefore we have conditions that $e_1 > e_2$, $b_1 > b_2$ and $e_2 + 1 \geq b_1$. The action will fix all segments in α except for Δ_1 and Δ_2, which will be replaced by one segment if $e_2 + 1 = b_1$, or $\Delta_3 = (b_2, \ldots, e_1)$ and $\Delta_4 = (b_1, \ldots, e_2)$ if $e_2 + 1 > b_1$. In the first case $\alpha \neq \beta$ trivially follows at the number of segments in the multisegments differ. In the second case, Δ_3 and Δ_4 will not be identical to Δ_1 and Δ_2, since $e_1 > e_2$ and $b_1 > b_2$. Thus in either case $\alpha \neq \beta$. $\square$

It now follows that we have a relation from the partial ordering of multisegments to the partial ordering of the rank triangles of their associated orbits.

Proposition 2.5.5. *Suppose that α is any multisegment and β is a multisegment such that $\alpha \leq \beta$. Let C and D be their respective corresponding conjugacy classes. Then $c_{i,j} \leq d_{i,j}$ for all i, j, where $c_{i,j}$ and $d_{i,j}$ are the associated ranks $r_{i,j}$ for the conjugacy classes C and D respectively. For ease of notation we will denote this as $C \leq D$.*

In other words, following the formation of the new multisegment β, then each of the associated ranks in the triangle of α will be less than or equal to the corresponding rank in the triangle associated to β.

Proof. It will be sufficient to show that following a single action on the multisegment α to create a new multisegment β will result in $C \leq D$ since any subsequent action will also have to satisfy this property. By **Proposition 2.5.4**, the multisegment will only change if the two segments in which the action is taken on have a preceding relation. Therefore if the action is taken on two non-preceding segments then it follows that $C = D$. Alternatively, let us now assume that the multisegment $\alpha \neq \beta$, then without loss of generality we can assume that the action was taken $\Delta_1 = \{b_1, \ldots, e_1\}$ and $\Delta_2 = \{b_2, \ldots, e_2\}$ where Δ_2 will precede Δ_1 ($e_1 > e_2$, $b_1 > b_2$ and $e_2 + 1 \geq b_1$.). The contributions to the ranks will only change for the integers contained in the segments thus we only need to show the ranks $c_{i,j} \leq d_{i,j}$ for $b_2 \leq i \leq j \leq e_1$ for the first two actions:

1. Firstly, let us assume $\Delta_1 \cap \Delta_2 \neq \emptyset$ and following the preceding relation the intersection is $\Delta_1 \cap \Delta_2 = \{b_1, \ldots, e_2\}$ and union $\Delta_1 \cup \Delta_2 = \{b_2, \ldots, e_1\}$. By construction Δ_1 and Δ_2 will contribute two to the rank triangle of C for each $c_{i,j}$ such that $b_1 \leq i \leq j \leq e_2$, and one for any other $c_{i,j}$ such that $b_1 \leq i \leq j \leq e_1$ or $b_2 \leq i \leq j \leq e_2$. Now studying the contributions of $\Delta_1 \cap \Delta_2$ and $\Delta_1 \cup \Delta_2$ in the rank triangle of D, we see that two will be contributed for any $d_{i,j}$ such that $b_1 \leq i \leq j \leq e_2$, otherwise, one will be contributed for any other $d_{i,j}$ in $b_2 \leq i \leq j \leq e_1$. Thus $c_{i,j} \leq d_{i,j}$ for all i, j following the union intersection.

2. To use the conjunction action the two segments must be such that $b_1 = e_2 + 1$. In the rank triangle of C, Δ_1 and Δ_2 will contribute one to the ranks $c_{i,j}$ such that $b_1 \leq i \leq j \leq e_1$ and $b_2 = e_1 + 1 \leq i \leq j \leq e_2$. Following the conjunction $\Delta_1 \cup \Delta_2$ will contribute one to the rank triangle of D for each $d_{i,j}$ such that $b_2 \leq i \leq j \leq e_1$, which thus encapsulates the contribution of α so $c_{i,j} \leq d_{i,j}$ following conjunction.

Therefore we have proved that for all three actions $c_{i,j} \leq d_{i,j}$, and hence whenever $\alpha \leq \beta$ then their associated rank triangles will be such that $c_{i,j} \leq d_{i,j}$ for all i, j. □

Likewise, it is also possible to study the relation from the partial ordering of the rank triangles to the partial ordering of the multisegments corresponding to their associated orbits.

Proposition 2.5.6. *Let C and D be the rank triangles associated to α and β with identical top rows. If C and D are both admissible and $C \leq D$, then*

$$\alpha \leq \beta.$$

Proof. Firstly, by assumption that the top rows of both C and D are the same, thus α and β will both have identical elements from which their multisegments are formed since they are admissible. Comparing α and β then we can now remove any multisegments in which α and β both contain. With the remaining segments in α and β, let us choose the longest segment in both containing the maximum element e then by construction the segment Δ_β chosen from β must be of greater length than that segment Δ_α from α since $C \leq D$. Also let b denote the base value of Δ_β, then there must exist at least one segment in α ending in b, and let us choose the longest such segment Δ'. This follows from the fact that a segment ending in b in β ensures the following property $D_{b-1,b} < D_{b,b}$.

Therefore by the initial assumptions we find

$$C_{b-1,b} \leq D_{b-1,b} < D_{b,b} = C_{b,b}.$$

If the union of these two segments form Δ_β then we are done. Otherwise, we need to show that there exists a segment Δ^* overlapping Δ_α with base value greater than b. So using a similar argument to above, let us denote the base value of Δ_α to be b_α then

$$C_{b-1,b_\alpha-1} \leq D_{b-1,b_\alpha-1} < D_{b_\alpha-1,b_\alpha-1} = C_{b_\alpha-1,b_\alpha-1}.$$

So there exists a segment containing $b_\alpha - 1$ in which the base value is greater than or equal to b. In fact, the base value of Δ^* is simply greater than b since Δ' is the longest segment with base value b and does not contain $b_\alpha - 1$. Also Δ_α is the longest segment ending in e, so the end of Δ^* is less than e. Once again we check if the union of all these segments form Δ_β at which point we are done. Otherwise, we have to recursively repeat this argument instead for the new segment Δ^* until the union of all of the segments found form Δ_β.

Thus we have shown the segment Δ_β can be constructed by the actions on α and hence we can compare α and β once again and remove any segments which they both contain. Note there may exist more than one segment (i.e. not only Δ_β) in common, since when union intersection is used it also creates a smaller segment which could also appear in both. Following this removal note C and D will once again remain admissible, and $C \leq D$ since any other segment generated in the formation of Δ_β will be of length less than the original segment. Thus by a recursive argument $\alpha \leq \beta$. $\square$

Therefore, **Proposition 2.5.5** and **Proposition 2.5.6** imply an overall relation between the partial ordering relation of multisegments and the rank triangles of their associated orbits.

Corollary 2.5.7. *Suppose that α and β are multisegments with respective corresponding conjugacy classes C and D. Then there exists a partial ordering on multisegments if and only if there exists a partial ordering on the corresponding conjugacy classes, that is,*

$$\alpha \leq \beta \ \textit{ if and only if } \ C \leq D.$$

Chapter 3 : Interpreting the Zelevinskii Involution

In this chapter, we study combinatorial interpretations of the Zelevinskii involution. These combinatorial methods will use the multisegment and rank triangle description previously introduced in Chapter 2 in order to compute the associated duals.

In Section 3.1, we will first introduce the Mœglin-Waldspurger algorithm for computing the Zelevinskii involution and find some preliminary results about this algorithm. The algorithm assembles segments of the dual by following a natural ordering based upon the precedes relation between segments. We will study how this natural ordering changes throughout the algorithm and the procedure in which segments of the dual are constructed. Given the dual multisegment is constructed using natural orderings then it may come as no surprise that we can define a network in order to represent these orderings. Furthermore, the surrounding literature shows that there exists a method using maximum flows through a network to implement the Zelevinskii involution.

In Section 3.2, we will generate a network based on the preceding relations defined by the original multisegment and show that the Mœglin-Waldspurger algorithm can be computed using this network. This will then allow us to show that there exists an abridged version of the Mœglin-Waldspurger algorithm for which we can add the extra condition that the segments must also initially precede. This work leads to an important consequence in **Corollary 3.2.12** that the dual multisegment can be constructed by exclusively using the initial natural ordering of the multisegments, which is a significant result in the thesis and will be instrumental in a number of the discussions that follow in Chapter 4.

Finally, we present a complete example in Section 3.3 for $G = GL_{16}(\mathbb{C})$ using the ideas from Chapter 2 and use the Network Computation of the Dual presented prior to it in the chapter to evaluate the dual.

3.1 Mœglin-Waldspurger Algorithm

Mœglin and Waldspurger's paper [13] detailing the implementation of the Zelevinskii involution using their eponymous algorithm includes a highly technical argument demonstrating their equivalence. The crux of this argument stems around the preceding condition first introduced in **Definition 2.5.3**.

The upcoming argument follows very closely [[13, Section II]], where we have translated both the language and the notation to follow that already in use. As previously discussed in the Quiver Representation Construction algorithm (See Section 2.3), given a multisegment $\alpha = \{\Delta_1, \ldots \Delta_n\}$ then the basis will be

$$\{\vec{e}_{\Delta,k} \mid \Delta \in \alpha, k \in \Delta\}.$$

Note that the dimensions of each space is thus determined by the total number of appearances of each k in segments of α.

We can then fix an endomorphism f of W by

$$f(\vec{e}_{\Delta_i,a}) = \begin{cases} \vec{e}_{\Delta_i,a+1}, & \text{if } a+1 \in \Delta_i; \\ 0, & \text{otherwise.} \end{cases}$$

This endomorphism given by f will be the element of V associated with the multisegment α, and we now seek to find an element g of V^* which commutes with f.

The use of the precedes relation can then be introduced by the following Lemma:

Lemma 3.1.1 ([13, Lemma II.4])**.** *Let $\alpha = \{\Delta_1, \ldots \Delta_n\}$ be a multisegment, f be the element of V associated with it. An element g of V^* commutes with f if and only if there exists a complex valued function on the set of pairs of segments contained in α, denoted by ν, satisfying:*

1. $\nu(\Delta_i, \Delta_j) = 0$*, if Δ_j does not precede Δ_i.*

2. *For all $i \in \{1, \ldots, n\}$ and $a \in \Delta_i$,*

$$g(\vec{e}_{\Delta_i,a}) = \sum_j \nu(\Delta_i, \Delta_j)\vec{e}_{\Delta_i,a-1},$$

summed over j such that $a-1 \in \Delta_j$.

The following proof is the translation of the proof provided in [13].

Proof. Let g be an element of V^*. We define a set of complex numbers $\{\mu(\Delta_i,a;\Delta_j,a-1)\}$ by the following formulas

$$g(\vec{e}_{\Delta_i,a}) = \sum_j \mu(\Delta_i,a;\Delta_j,a-1)\vec{e}_{\Delta_i,a-1},$$

summed over all j such that $a-1 \in \Delta_j$, for all $i \in \{1,\dots,n\}$ and $a \in \Delta_i$. It is convenient here to assume that $\mu(\Delta_i,a;\Delta_j,a-1)$ is zero if $a \notin \Delta_i$ or $a-1 \notin \Delta_j$. We have for all $i \in \{1,\dots,n\}$ and $a \in \Delta_i$:

$$(fg-gf)(\vec{e}_{\Delta_i,a}) = \sum_j \left[\mu(\Delta_i,a;\Delta_j,a-1) - \mu(\Delta_i,a+1;\Delta_j,a)\right]\vec{e}_{\Delta_j,a},$$

summed over all j such that $a \in \Delta_j$.

Hence f and g commute if and only if the following relations are satisfied for all $i,j \in \{1,\dots,n\}$, $a \in \Delta_i \cap \Delta_j$:

(1) If $a-1 \in \Delta_j$ and $a+1 \in \Delta_i$, then

$$\mu(\Delta_i,a;\Delta_j,a-1) = \mu(\Delta_i,a+1;\Delta_j,a).$$

(2) If $a-1 \in \Delta_j$ and $a+1 \notin \Delta_i$, then

$$\mu(\Delta_i,a;\Delta_j,a-1) = 0.$$

(3) If $a+1 \in \Delta_i$ and $a-1 \notin \Delta_j$, then

$$\mu(\Delta_i,a+1;\Delta_j,a) = 0.$$

Suppose that f and g commute and define ν as follows:

(a) $\nu(\Delta_i,\Delta_j) = 0$, if for all elements a of Δ_i, $a-1$ does not belong to Δ_j.

(b) $\nu(\Delta_i,\Delta_j) = \mu(\Delta_i,a;\Delta_j,a-1)$, if there exists one and only one element, denoted a of Δ_i, such that $a-1$ belongs to Δ_j, and if Δ_j does not contain the element a. According to (3), $\nu(\Delta_i,\Delta_j) = 0$ if Δ_j does not precede Δ_i.

(c) Let us now fix a,Δ_i,Δ_j such that a is an element of Δ_i and Δ_j, and $a-1$ belongs to Δ_j. Let c (resp. d) be the greatest integer such that $(a-c)$ (resp. $(a+d)$) belong to Δ_i (resp. Δ_j), that is,

$c = a - b_{\Delta_i}$ (resp. $d = e_{\Delta_j}$-a) where b_Δ and e_Δ denote the base and end values respectively of the segment Δ. Thanks to (1) and (2), then from (1) and (3) we obtain the following additional conditions for μ:

(4)

$$\mu(\Delta_i, a; \Delta_j, a-1) = \begin{cases} \mu(\Delta_i, a+d+1; \Delta_j, a+d), & \text{if } a+d+1 \in \Delta_i, \\ 0, & \text{otherwise.} \end{cases}$$

(5)

$$\mu(\Delta_i, a; \Delta_j, a-1) = \begin{cases} \mu(\Delta_i, a-c; \Delta_j, a-c-1), & \text{if } a-c-1 \in \Delta_j, \\ 0, & \text{otherwise.} \end{cases}$$

Thus $\mu(\Delta_i, a; \Delta_j, a-1) = 0$ if Δ_j does not precede Δ_i, and according to (4) and (5) coincides with $\mu(\Delta_i, a'; \Delta_j, a'-1)$ for all a' belonging to Δ_i and Δ_j such that $a'-1$ belongs to Δ_j.

We then set

$$\nu(\Delta_i, \Delta_j) = \mu(\Delta_i, a; \Delta_j, a-1).$$

It is clear that we have

$$g(\vec{e}_{\Delta_i,a}) = \sum_j \nu(\Delta_i, \Delta_j)\vec{e}_{\Delta_i,a-1},$$

summed over j such that $a-1 \in \Delta_j$.

This finishes the proof of the direct assertion of the lemma and the converse immediately follows thanks to the calculation of $(fg - gf)$. □

Therefore any element g of V^* must hence be constructed from those segments that precede, otherwise the element will not commute with f. The intricate nature of the use of this preceding relation in the Mœglin-Waldspurger algorithm allows us to find the associated dual multisegment without actually finding a representative. The equivalency of these representatives generated by the Zelevinskii involution and the multisegments by the Mœglin-Waldspurger algorithm is demonstrated in [13].

We can now begin to introduce the Mœglin-Waldspurger algorithm which will compute the multisegment associated to the dual orbit from the original multisegment. The algorithm will use the previously defined preceding relation between segments given in **Definition 2.5.3**.

Algorithm : Mœglin-Waldspurger [13] Given a multisegment α with maximum value e then we can compute the multisegment $\tilde{\alpha}$ associated to the dual orbit as follows:

1. Let $m = e$ be the maximum value in the multisegment and set Δ_m to be the shortest segment whose maximal value is m.
2. If there does not exist a segment that precedes Δ_m whose maximal value is $m-1$, then go to step 5.
3. Amongst the segments that precede Δ_m whose maximal value is $m-1$, select Δ_{m-1} to be the shortest such segment.
4. Set $m := m-1$ and return to step 2.
5. For each segment Δ_i for $m \leq i \leq e$ remove the maximal value i from this segment. Following the removal of these end values, let us denote the new multisegment to be α'.
6. The dual segment formed will be $\Delta' = (m, \ldots, e)$.

Generating the segment Δ' will be from here forward referred to as a single iteration of the algorithm. To find the complete dual multisegment $\tilde{\alpha}$ one will need to continue this process recursively using

$$\tilde{\alpha} = \left\{\Delta', \widetilde{(\alpha')}\right\}.$$

Mœglin and Waldspurger then prove in [13, Theorem 13] that $\tilde{\alpha}$ will be equal to the multisegment of the dual representation found by the Zelevinskii involution.

Example 3.1.2. Let us consider the multisegment given by

$$\alpha = \{(1), (1,2), (2,3), (2,3), (3), (3,4)\}.$$

We can compute the dual multisegment $\tilde{\alpha}$ using the Mœglin-Waldspurger algorithm as shown below. At each iterative step the segments chosen for each Δ_i are labelled in red and their end point has a red box around it. Once removed by the algorithm the end integer will then be greyed out. The progress of the construction of the dual multisegment at the start of the iteration is then shown below the iteration, and each constructed segment is labelled depending on the iteration it was generated in.

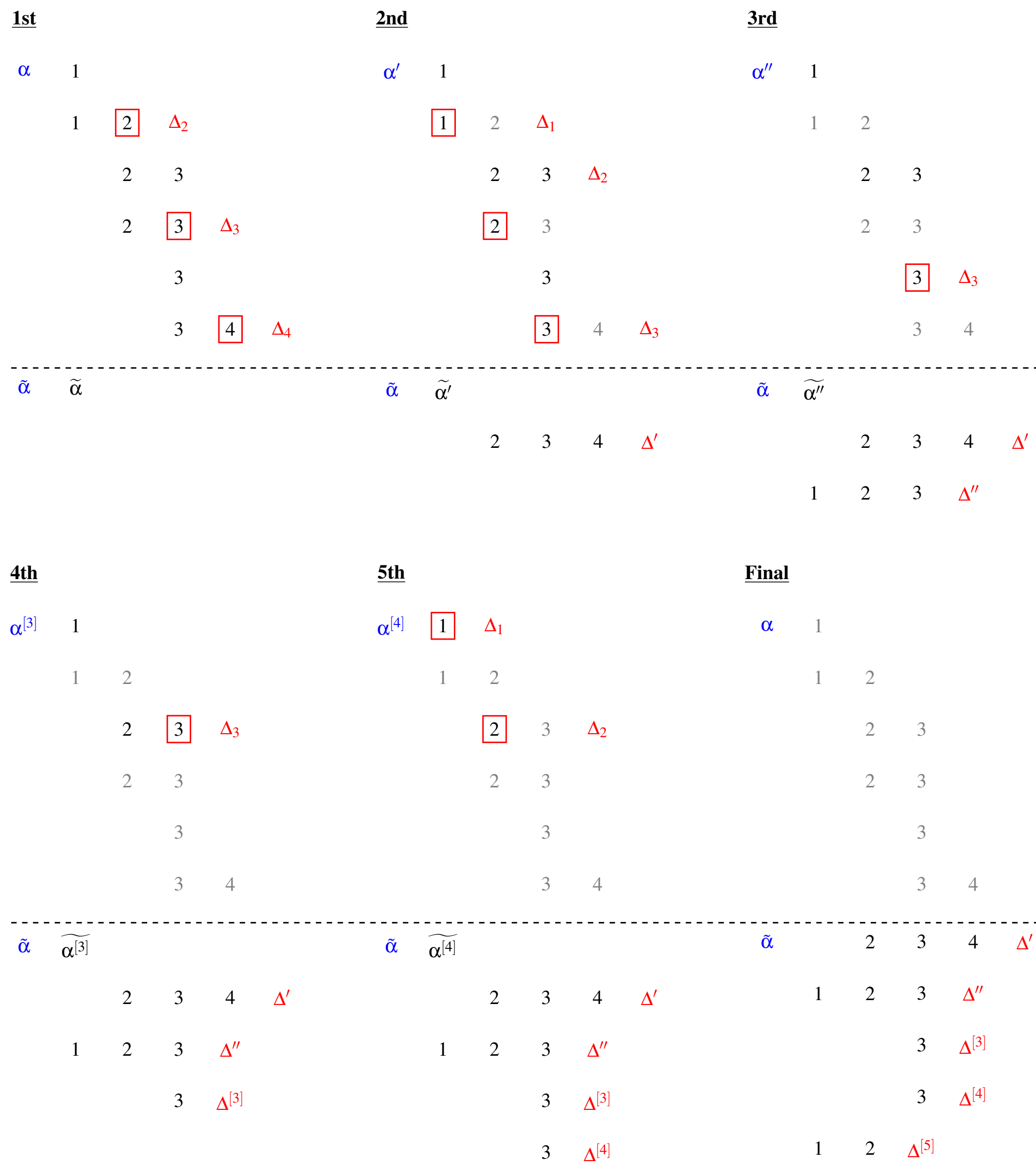


Figure 3.1: Implementing the Mœglin-Waldspurger algorithm.

Therefore we have found that the multisegment associated to the dual is given by

$$\tilde{\alpha} = \{(2,3,4),(1,2,3),(3),(3),(1,2)\}.$$

We can define a special family of multisegments:

Definition 3.1.3. We say that a multisegment α is *self-dual* if it has the property

$$\alpha = \tilde{\alpha}.$$

We will now present a couple more examples of highly structured multisegments and how the Mœglin-Waldspurger algorithm can be computed on them. Note that we will use a single diagram for each example - the blue arrows will indicate the segments generated and be labelled with the iteration for which the segment was generated on.

Example 3.1.4. The first example is a *simple multisegment* (See **Definition 4.2.6**) for which each segment has the same length and the end values reduce by one each time.

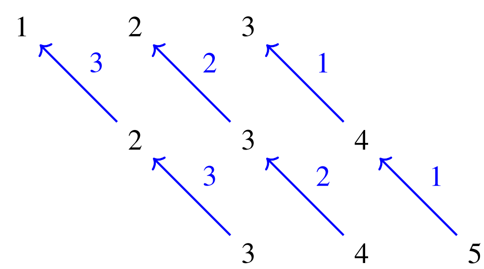


Figure 3.2: The Mœglin-Waldspurger algorithm on a simple multisegment.

The dual of $\{(1,2,3),(2,3,4),(3,4,5)\}$ is $\{(1,2,3),(2,3,4),(3,4,5)\}$, hence this multisegment has an additional property of being self-dual.

Example 3.1.5. This example is a *ladder multisegment* (See **Definition 4.2.12**) for which there exists a complete ordering of the segments based around their base and end values.

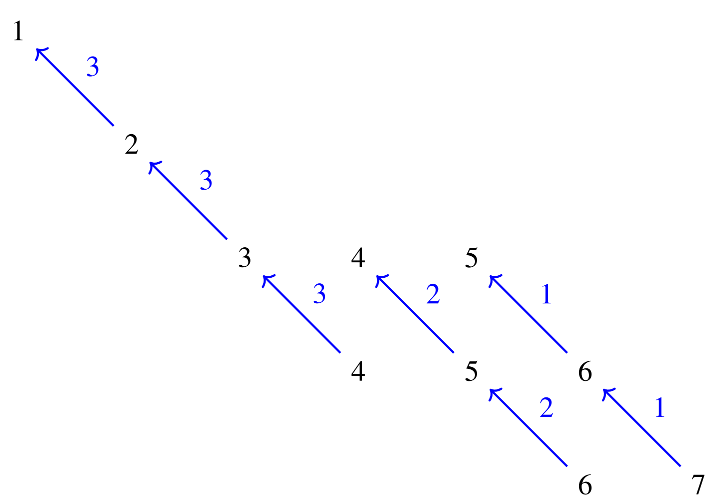


Figure 3.3: The Mœglin-Waldspurger algorithm on a simple multisegment.

The dual of $\{(1),(2),(3,4,5),(4,5,6),(6,7)\}$ is $\{(1,2,3,4),(4,5,6),(5,6,7)\}$.

Further studies of both simple multisegments and ladder multisegments will be found in **Chapter 4**.

Following the computation of the algorithm in both **Example 3.1.4** and **Example 3.1.5** then one may be inclined to believe that we can always partition the multisegments into sub-multisegements of the form shown in **Example 3.1.5**. In other words, create sub-multisegements which contain firstly the shortest segment containing the maximal value not chosen in a sub-multisegment and then segments which precede the previously chosen segment and are both shortest and end with the highest value. Following this we can then compute the duals of sub-multisegments independently to form the dual of the overall multisegment. In general this will not be true as we will see in the next example. **Section 4.1** will be devoted to this study of partitioning multisegments into sub-multisegments.

Example 3.1.6. Let us consider the multisegment given by

$$\alpha = \{[0,4],[1,2],[2,3],[3,5]\}.$$

If we partition the multisegment into sub-multisegments using the procedure described above, then we generate two sub-multisegments

$$\alpha_1 = \{[0,4],[3,5]\} \text{ and } \alpha_2 = \{[1,2],[2,3]\},$$

which corresponds to the dual multisegment

$$\{\tilde{\alpha_1}, \tilde{\alpha_2}\} = \{[0],[1],[2,3],[3,4],[4,5],[1,2],[2,3]\}.$$

However the Mœglin-Waldspurger algorithm instead finds the dual to be

$$\tilde{\alpha} = \{[0],[1],[2],[3],[1,3],[2,4],[4,5]\},$$

which is clearly not the same.

There also exists an analogous algorithm to the Mœglin-Waldspurger which instead constructs the multisegments starting from the smallest value appearing in any segment of α. This analogous description is often used in the literature surrounding this topic. However in general we will only need to consider the original algorithm.

Algorithm : Alternative Mœglin-Waldspurger [13] Let us define ζ to be a function which computes the dual of a multisegment α by an inductive method. To simplify the notation we will now represent the segment $(i, i+1, \ldots, j)$ by $[i, j]$, or if the segment is a singleton (i) then $[i]$. Thus each segment in α will hence be of the form $[i, j]$ for $1 \leq i \leq j$. We will define the multiplicity of the segment $[i, j]$ in α by $m_{i,j}$ and the *weights of the multisegment* to be $(g_1, g_2 \ldots)$, where g_i for all i are simply given by the multiplicity of i in α. So $\zeta(\alpha)$ is found by the following inductive process:

1. Set $i_1 = \min\{i \mid g_i \neq 0\}$.

2. Let us set

$$j_1 = \min\{j \mid m_{i_1,j} \neq 0\} \text{ and } j_{t+1} = \min\{j \mid j > j_t, m_{i_1+t,j} \neq 0\}, \text{ (for } t = 1, \ldots, p-1).$$

   Note the sequence terminates once j_{p+1} does not exist, that is, $m_{i_1+p,j} = 0$ for all $j_p < j$.

3. Set

$$\begin{aligned} \alpha^{(n+1)} = \alpha^{(n)} &- [i_1, j_1] - [i_1+1, j_2] - \cdots - [i_1+p-1, j_p] \\ &+ [i_1+1, j_1] + [i_1+2, j_2] + \cdots + [i_1+p, j_p], \end{aligned}$$

   where $[i, j] = 0$ if $i > j$.

4. The dual multisegment then becomes

$$\zeta\left(\alpha^{(n)}\right) = \zeta\left(\alpha^{(n+1)}\right) + [i_1, i_1+p-1].$$

5. Repeat for each $\alpha^{(n)}$ until $\alpha^{(n)}$ is empty.

To illustrate the fact that both algorithms compute the same multisegments associated to the dual, let us repeat **Example 3.1.2** using the alternative Mœglin-Waldspurger algorithm.

Example 3.1.7. Let consider the multisegment

$$\alpha = [1] + [1,2] + [2,3] + [2,3] + [3] + [3,4].$$

We can compute the dual multisegment $\tilde{\alpha}$ using the alternative Mœglin-Waldspurger algorithm as follows:

1st: Firstly, the multiplicities are given by $(g_1, g_2, g_3, g_4) = (2,3,4,1)$, so set $i_1 = 1$ then we find that $(j_1, j_2, j_3) = (1,3,4)$. This results in $\zeta(\alpha) = \zeta(\alpha') + [1,3]$, where

$$\begin{aligned}\alpha' &= \alpha - [1] - [2,3] - [3,4] + [2,1] + [3] + [4],\\ &= [1,2] + [2,3] + [3] + [3] + [4].\end{aligned}$$

2nd: The multiplicities are given by $(g_1, g_2, g_3, g_4) = (1,2,3,1)$, so set $i_1 = 1$ then we find that $(j_1, j_2) = (2,3)$. This results in $\zeta(\alpha) = \zeta(\alpha'') + [1,3] + [1,2]$, where

$$\begin{aligned}\alpha'' &= \alpha' - [1,2] - [2,3] + [2] + [3],\\ &= [2] + [3] + [3] + [3] + [4].\end{aligned}$$

3rd: The multiplicities are given by $(g_1, g_2, g_3, g_4) = (0,1,3,1)$, so set $i_1 = 2$ then we find that $(j_1, j_2, j_3) = (2,3,4)$. This results in $\zeta(\alpha) = \zeta(\alpha^{[3]}) + [1,3] + [1,2] + [2,4]$, where

$$\alpha^{[3]} = \alpha'' - [2] - [3] - [4] + [3,2] + [4,3] + [5,4] = [3] + [3].$$

4th: The multiplicities are given by $(g_1, g_2, g_3, g_4) = (0,0,2,0)$, so set $i_1 = 3$ then we find that $j_1 = 3$. This results in $\zeta(\alpha) = \zeta(\alpha^{[4]}) + [1,3] + [1,2] + [2,4] + [3]$, where

$$\alpha^{[4]} = \alpha^{[3]} - [3] + [4,3] = [3].$$

5th: Finally, the multiplicities are given by $(g_1, g_2, g_3, g_4) = (0,0,1,0)$, so set $i_1 = 3$ then we find that $j_1 = 3$. This results in $\zeta(\alpha) = \zeta(\alpha^{[5]}) + [1,3] + [1,2] + [2,4] + [3] + [3]$ where

$$\alpha^{[5]} = \alpha^{[4]} - [3] + [4,3] = \emptyset.$$

Thus the algorithm terminates after this iteration.

Therefore we have found that the multisegment associated to the dual is given by

$$\zeta(\alpha) = [1,3] + [1,2] + [2,4] + [3] + [3],$$

and hence equal to that generated by the Mœglin-Waldspurger algorithm as shown in **Example 3.1.2**.

Theorem 3.1.8. *Given a multisegment* α *then both the Mœglin-Waldspurger and Alternative Mœglin Waldspurger algorithms compute the same dual multisegment of* α*.*

Proof. To prove that the two algorithms are equivalent, we will simply show that at each inductive stage of the algorithm they construct the same dual segment and start the following iteration at the same point. This implies that the overall dual multisegments computed by each of algorithms will thus be equal.

Firstly, let α be a multisegment. Then we will define α^- to be such that the segment $[i, j] \in \alpha$ if and only if the segment $[-j, -i] \in \alpha^-$. We can do this since the construction of the multisegments is simply assigning an index to each of the eigenvalues.

Let us now compute the first iteration of the alternative Mœglin Waldspurger algorithm on α:

1. Set $i_1 = \min\{i \mid g_i \neq 0\}$.

2. Then let us assume that j_{p+1} is the first $j_t > j_1$ which does not exist. So we have the set $(j_1, \ldots, j_p)$ which has been arbitrarily chosen and satisfies the conditions defined in the algorithm.

3. So

$$\begin{aligned}\alpha' = \alpha - [i_1, j_1] - [i_1+1, j_2] - \cdots - [i_1+p-1, j_p] \\ + [i_1+1, j_1] + [i_1+2, j_2] + \cdots + [i_1+p, j_p].\end{aligned}$$

4. The dual multisegment can be defined recursively by

$$\zeta(\alpha) = \zeta(\alpha') + [i_1, i_1+p-1],$$

so $[i_1, i_1+p-1] \in \zeta(\alpha)$.

Further, let us now compute the first iteration of the Mœglin Waldspurger algorithm on α^-:

(1) $-i_1$ will largest value in α^- since i_1 was the smallest value in α. Also $[-j_1, -i_1]$ is the shortest segment in α^- ending with $-i_1$, because by construction and above $[i_1, j_1]$ is the shortest segment starting with i_1. So let $\Delta_{-i_1} = [i_1, j_1]$.

(2-4) For each $n = 1, \ldots, p-1$, there exists a segment $[i_{n+1}, j_{n+1}]$ for which $i_{n+1} = i_n + 1$ and $j_{n+1} > j_n$ by the alternative Mœglin Waldspurger algorithm on α. Note each of these segments will be the shortest possible segment starting with i_{n+1} which satisfy these conditions. So let $\Delta_{-i_{n+1}} = [-j_{n+1}, -i_{n+1}]$. Then by above we have $-i_{n+1} = -i_n - 1 < -i_n$ and $-j_n > -j_{n+1}$. By the alternative Mœglin Waldspurger algorithm on α, each $i_m \leq j_m$ for $m = 1, ..., p$. Therefore for each $n = 1, ..., p-1$;

$$-j_n \leq -i_n = -i_{n+1} + 1,$$

so we have satisfied the conditions for which $\Delta_{-i_{n+1}}$ precedes Δ_{-i_n}, and $\Delta_{-i_{n+1}}$ is the shortest such segment in each case. Note that there will be no segment that precedes Δ_{-i_p}, because in the alternative Mœglin Waldspurger algorithm there does not exist a segment which starts at $i_1 + p$ and is such that $j_{p+1} > j_p$ so the preceding conditions will not be met.

(5) If we now remove each i from Δ_i for $i = -i_1 - p + 1, \ldots, -i_1$, to do this we can use

$$\begin{aligned}(\alpha^-)' = \alpha^- &- [-j_1, -i_1] - [-j_2, -i_1 - 1] - \cdots - [-j_p, -i_1 - p + 1] \\ &+ [-j_1, -i_1 - 1] + [-j_2, -i_1 - 2] + \cdots + [-j_p, -i_1 - p].\end{aligned}$$

(6) Let us add the segment $[-i_1 + p - 1, -i_1]$ to $\zeta(\alpha^-)$.

It is clear that both algorithms compute the same dual segment since $[-i_1 - p + 1, -i_1]$ in α^- corresponds to $[i_1, i_1 + p - 1]$ in α. What remains is to prove that the starting point of the next iterations are equivalent.

To prove this let us compute

$$\begin{aligned}
\left[(\alpha^-)'\right]^- &= \Big[\alpha^- - [-j_1, -i_1] - [-j_2, -i_1 - 1] - \cdots - [-j_p, -i_1 - p + 1] \\
&\qquad + [-j_1, -i_1 - 1] + [-j_2, -i_1 - 2] + \cdots + [-j_p, -i_1 - p]\Big]^-, \\
&= (\alpha^-)^- - [-j_1, -i_1]^- - [-j_2, -i_1 - 1]^- - \cdots - [-j_p, -i_1 - p + 1]^- \\
&\qquad + [-j_1, -i_1 - 1]^- + [-j_2, -i_1 - 2]^- + \cdots + [-j_p, -i_1 - p]^-, \\
&= \alpha - [i_1, j_1] - [i_1 + 1, j_2] - \cdots - [i_1 + p - 1, j_p] \\
&\qquad + [i_1 + 1, j_1] + [i_1 + 2, j_2] + \cdots + [i_1 + p, j_p], \\
&= \alpha'.
\end{aligned}$$

Therefore both algorithms will iteratively compute the same duals. □

3.1.1 Understanding the Mœglin-Waldspurger Algorithm

This subsection will be dedicated to studying the process in which the Mœglin-Waldspurger algorithm selects end values to form segments in the dual multisegment. We will also investigate how the preceding relations change, and prove a number of facts about how the algorithm proceeds. For ease of notation, given a segment $\Delta = (b, b+1, \ldots, e-1, e)$, then we say the *base* of Δ is b and the *end* of Δ is e. That is, the base of a segment corresponds to the smallest integer and the end of the segment to the largest integer in the segment. Therefore the length of the segment Δ will be given by $e - b + 1$.

Proposition 3.1.9. *During each iteration of the Mœglin-Waldspurger algorithm the preceding segments will be chosen in increasing length.*

Proof. Let m be the integer chosen from the segment Δ_m by the Mœglin-Waldspurger. If m is not the base of the segment generated by the algorithm, then there exists a segment Δ_{m-1} that precedes Δ_m with end value $m-1$. Let b_m denote the base value of the segment Δ_m, then by the precedes condition $b_{m-1} < b_m$, which implies that

$$m - b_m + 1 < m - b_{m-1} + 1,$$

and

$$m - b_m + 1 \leq (m-1) - b_{m-1} + 1.$$

Thus the length of the preceding segments will be chosen in increasing length. $\square$

As a consequence, throughout each iteration of the Alternative Mœglin-Waldspurger algorithm the proceeding segments will also be chosen in increasing length.

The preceding relations will adjust following each iteration of the algorithms, so the preceding relations in α' are not strictly those from α. In light of this **Lemma 3.1.1** may seem counterintuitive, since the precedes relation initially defined by Mœglin and Waldspurger in [13] will remain fixed when computing g of V^*.

The relations change with each iteration of the algorithm for both versions of the Mœglin-Waldspurger algorithm.

Proposition 3.1.10. *Let $\Delta_1 = [b_1, e_1]$ and $\Delta_2 = [b_2, e_2]$ be two segments in the multisegment with no preceding relation between them. In order to create a preceding relation (Δ_1 precedes Δ_2 in α') between them after applying one iteration of the Mœglin-Waldspurger algorithm, then the following conditions must be satisfied:*

1. *The segments Δ_1 and Δ_2 must be such that $e_1 = e_2$ and $b_1 > b_2$.*
2. *There must exist a segment $\Delta_3 = [b_3, e_1 + 1]$ for which $e_1 + 1$ is chosen from Δ_3 in the same iteration as e_1 from Δ_2, hence $b_2 < b_3 \leq b_1$.*

Proof. Firstly, we are only considering a single iteration of the Mœglin-Waldspurger algorithm so there can only be a singular change to the end values of any segments. This will not impact the base values so it follows that $b_1 > b_2$ for any preceding condition to be formed. Also following the iteration the end value of Δ_1 must be greater than the end value of Δ_2. This must be an aspect that is changed during the iteration since this condition cannot already be true otherwise Δ_1 would initially precede Δ_2. Additionally the end value of Δ_2 cannot be greater than that of Δ_1 since this would require more than one iteration to generate a preceding relation. By process of elimination the end values must be equal.

In order to now generate a precedes condition the e_1 must be chosen from Δ_2 before the e_1 is chosen from Δ_1. Note if there are no preceding segments of both Δ_1 and Δ_2 then e_1 would be chosen from Δ_1 since $b_1 > b_2$. Likewise if there exists a segment that precedes Δ_1 (and hence also Δ_2) then e_1 would be chosen from Δ_1. It follows that there must instead exist a segment Δ_3 that only precedes Δ_2. Note we are only considering a single iteration so the end value of Δ_3 must be $e_1 + 1$ and since it precedes Δ_2 but not Δ_1 then $b_2 < b_3 \leq b_1$. □

For example $\Delta_1 = (2,3,4)$, and $\Delta_2 = (1,2,3,4)$. By construction there is no preceding relation on this iteration, however if the end value (4) of Δ_2 is removed during this next iteration then Δ_2 will be such that it precedes Δ_1, that is, $\Delta_2' = (1,2,3)$ will precede $\Delta_1 = (2,3,4)$. However the value (4) of Δ_2 is only removed before the value (4) of Δ_1 if there exists a segments Δ_3 which precedes Δ_2 but not Δ_1 and selects the value in the same iteration, that is, $\Delta_3 = (2,3,4,5)$. Following a single iteration, we will therefore have $\Delta_1 = (2,3,4)$, $\Delta_2' = (1,2,3)$ and $\Delta_3' = (2,3,4)$, so a new preceding relation has been generated since Δ_2' will now be preceded by both Δ_1 and Δ_3'.

Proposition 3.1.11. *Let $\Delta_1 = [b_1, e_1]$ and $\Delta_2 = [b_2, e_2]$ be two segments in the multisegment such that Δ_1 precedes Δ_2. In order to destroy this preceding relation (Δ_1 does not precede Δ_2 in α') after applying one iteration of the Mœglin-Waldspurger algorithm, then the following conditions must be satisfied:*

1. *The segments Δ_1 and Δ_2 must be such that $e_1 - 1 = e_2$.*

2. *There must exist a segment $\Delta_3 = [b_3, e_2]$ for which e_1 is chosen from Δ_1 in the same iteration as $e_1 - 1$ from Δ_3, hence $b_2 \leq b_3 < b_1$.*

Proof. Firstly, we are only considering a single iteration of the Mœglin-Waldspurger algorithm so there can only be a singular change to the end values of any segments. This will not impact the base values so $b_1 > b_2$ will remain satisfied and the condition $e_1 > e_2$ must be broken. Therefore $e_1 > e_2$ must initially be true be able to be broken by a single iteration, hence $e_1 - 1 = e_2$. In order to now destroy the precedes condition e_1 must be chosen from Δ_1 in a separate iteration to the $e_1 - 1$ from Δ_2. It follows that there must instead exist a segment Δ_3 that is preceded by Δ_1 and is chosen before Δ_2. Note Δ_3 will only be chosen following Δ_1 before Δ_2 if it precedes Δ_1 and has length less than or equal to Δ_2 so $b_2 \leq b_3 < b_1$. □

For example $\Delta_1 = (2,3,4)$, and $\Delta_2 = (0,1,2,3)$. By construction there initially exists a preceding relation on this iteration. However if the end value(s) (4) from Δ_1 are removed by the algorithm but the end value of Δ_2 (4) remains fixed then the end values of both multisegments will eventually become equal. If this happens then during this next iteration there will be no such preceding relation. However for this to happen there must exist a segment Δ_3 which precedes Δ_1, has an end value of (3) and has a base value greater than or equal to the base value of Δ_2, thus we could take $\Delta_3 = (1,2,3)$. Following a single iteration, we will therefore have $\Delta_1' = (2,3)$, $\Delta_2 = (0,1,2,3)$ and $\Delta_3' = (1,2)$, so the initial preceding relation between Δ_1 and Δ_2 will no longer exist.

As we will see in the next proposition the creation and destruction of preceding relations will only have an influence when the maximum value in the multisegment reduces during the computation of the Mœglin-Waldspurger algorithm.

Proposition 3.1.12. *Let Δ be any segment of a multisegment α. When carrying out the Mœglin-Waldspurger algorithm on α, then it is not possible that both of the integers $i-1$ and i of Δ are chosen to be in dual segments with the same end value.*

Proof. Let e_α be the maximum value of our multisegment α. When we compute the dual of α using the Mœglin-Waldspurger algorithm then all the segments of the dual ending in e_α must be computed before any segment ending in a value less than e_α. The iterations computing these segments will also choose Δ_{e_α} in order of shortest to largest. Then any preceding segments chosen during each iteration must then be such that they are chosen with increasing length in order for them to precede. Now given that the Δ_{e_α} must also increase with each ongoing iteration then in turn $\Delta_{e_\alpha-1}, \Delta_{e_\alpha-2}, \dots$ will also increase in length every iteration.

To show this let us use double induction on the pair (i,k) for $i \geq 1$ and $k \geq 0$, where i denotes the ith iteration of the Mœglin-Waldspurger algorithm for which a segment ending e_α is generated and k denotes the segment $\Delta_{e_\alpha-k}$ which is chosen. For ease of notation we will say that $\Delta^i_{e_\alpha-k}$ (if it exists) is the segment in which $e_\alpha - k$ is chosen from on the ith iteration.

We already know by **Proposition 3.1.9** that when $i=1$ $\Delta^1_{e_\alpha-k}$ is longer than $\Delta^1_{e_\alpha-(k-1)}$ for all $k>0$. Additionally when $k=0$ $\Delta^i_{e_\alpha}$ is longer than $\Delta^i - 1_{e_\alpha}$ for all $i>1$, since the segments ending in the maximum value are chosen in ascending order of length and new segments ending in e_α cannot be created since this would contradict the fact that e_α is the maximum value.

Let us now assume that it is true for $i = n$ and $k = m-1$ (for $n \geq 1, m > 0$) and $i = n-1$ and $k = m$ (for $n > 1, m \geq 0$). Now we need to prove that for $i = n$ and $k = m$ that $\Delta^n_{e_\alpha - m}$ is longer than $\Delta^{n-1}_{e_\alpha - m}$. If we assume that instead $\Delta^n_{e_\alpha - m}$ is shorter than $\Delta^{n-1}_{e_\alpha - m}$. Recall that the Mœglin-Waldspurger always chooses the shortest segment for any end value. If both $\Delta^n_{e_\alpha - m}$ and $\Delta^{n-1}_{e_\alpha - m}$ are present and end with $e_\alpha - m$ during the $n-1$, then this would imply that $\Delta^{n-1}_{e_\alpha - m}$ is shorter than or the same length. Therefore if $\Delta^n_{e_\alpha - m}$ is to be shorter than $\Delta^{n-1}_{e_\alpha - m}$ then it must be newly created and hence $e_\alpha - (m-1)$ must have been end value of $\Delta^n_{e_\alpha - m}$ on the $(n-1)$th iteration, that is, $\Delta^{n-1}_{e_\alpha - (m-1)}$.

By the inductive hypothesis ($i = n$ and $k = m-1$), $\Delta^n_{e_\alpha - (m-1)}$ is longer than $\Delta^{n-1}_{e_\alpha - (m-1)}$. If the end value $e_\alpha - (m-1)$ of $\Delta^{n-1}_{e_\alpha - (m-1)}$ is removed then it will become $\Delta^n_{e_\alpha - m}$ and hence will be strictly shorter than $\Delta^n_{e_\alpha - (m-1)}$. This will lead to a contradiction since **Proposition 3.1.9** states that the preceding segments which are chosen must be in increasing order of length. Therefore we have proved that as i increases then the length of the segment $\Delta^i_{e_\alpha - k}$ must also increase for $k \geq 0$.

During each iteration the lengths of the new segments generated by removing the end values of $\Delta_{e_\alpha}, \Delta_{e_\alpha - 1}, \ldots$ will become shorter. It will therefore not be possible for a segment to be chosen twice for the same end value e_α, since segments must be chosen in increasing order of length during both the iteration and for the respective end values. □

Corollary 3.1.13. *Whilst computing all segments ending in the maximum value using the Mœglin-Waldspurger algorithm the preceding relations can remain the same. That is, if you use the new precedes relations of* α' *or simply inherit those from* α *the algorithm will make the same choices.*

Corollary 3.1.14. *Whilst computing all segments with the minimum value being the base value using the alternative Mœglin-Waldspurger algorithm the preceding relations can remain the same.*

Proposition 3.1.15. *Whilst computing all segments corresponding to the maximum value using the Mœglin-Waldspurger algorithm each segment can only be used at most once, and for each* $i \leq e_\alpha$ *the segment* Δ_i *must be chosen in increasing length as the iterations go on.*

Proof. This follows from the proof of **Proposition 3.1.12** □

Corollary 3.1.16. *Each iteration of the Mœglin-Waldspurger algorithm chooses the longest possible dual segment ending with the maximum value remaining in the multisegment.*

The algorithm can therefore be implemented one end value at a time and through the construction of a table of all segments categorised by their end values in increasing order of length. Then each segment corresponding to the maximum value will be chosen in order and any segment chosen from a subsequent row must then have length greater than or equal to the previous segment. If we relook at **Example 3.1.2** then the original multisegment has maximum value 4 and the table would be as follows:

Table 3.1: Implementing the Mœglin-Waldspurger algorithm on maximum value 4

End Values (n)	Segments ending in n		
4	[3,4]		
3	[3]	[2,3]	[2,3]
2	[1,2]		
1	[1]		

Following the computations, any segment which was chosen will have its end values removed and be moved down a row if it still exists.

3.2 Network Approach

In this section we will study how networks can be used in the implementation of the Zelevinskii involution. Following this, we look into what this network theoretic approach means for the Mœglin-Waldspurger algorithm, which will in turn allow us to present an abridged version of the Mœglin-Waldspurger algorithm that restricts the choices at each stage. These restrictions and the graph theoretic description will be instrumental in the main results of the thesis presented in Chapter 4.

The decision to study the network description of the Zelevinskii Involution follows from the surrounding literature, mainly Zelevinskii's work with Knight. In their paper [9], Knight and Zelevinskii use the results of Poljak's theorem for describing the maximal rank of the p^{th} power of matrices with a given pattern [15]. This allows for the formation of a closed form solution for finding the rank triangle associated to the dual multisegment. Let b and e respectively denote the minimum and maximum integers of the multisegment α. Knight and Zelevinskii construct a graded vector space

$$E = E_b \oplus E_{b+1} \oplus \cdots \oplus E_{e-1} \oplus E_e,$$

comparable to the vector space shown in **Figure 2.1** in which the dimension of each E_i is equal to the associated weight g_i (the multiplicity of i in the multisegment α). Each $f \in V$ will be nilpotent and hence a Jordan decomposition can be chosen consisting of Jordan graded cells. Each cell associated to a segment $[i, j]$ and consists of f-invariant subspace of V

$$L_i \oplus L_{i+1} \oplus \cdots \oplus L_{j-1} \oplus L_j,$$

where $\dim(L_k) = 1$, $L_k \subset E_k$, and $f(L_k) = L_{k+1}$ for $k = i, \ldots, j-1$.

Following on from the Jordan decomposition, we then define $T_{i,j}$ for all (i, j) to be the set of all maps

$$v : [b,i] \times [j,e] \to [i, j]$$

such that $v(k,l) \leq v(k',l')$ whenever $k \leq k'$ and $l \leq l'$. These maps will simply correspond to all possible chains that can be made when constructing g in the Zelevinskii involution. Knight and Zelevinskii then use the maps to formulate the closed form solution as follows:

Theorem 3.2.1 ([9, Theorem 1.2])**.** *For any multisegment $\alpha = (m_{i,j})_{b \leq i \leq j \leq e}$, we have*

$$\widetilde{r_{i,j}} = \min_{v \in T_{i,j}} \sum_{(k,l) \in [b,i] \times [j,e]} m_{v(k,l)+k-i, v(k,l)+l-j}.$$

The proof then follows by interpreting the isomorphism classes of the associated quiver representation as graded nilpotent operators previously discussed. A network is then formed using this description and the maps $T_{i,j}$ are pivotal in the formation of the directed edges. The maximum-flow and minimum cut theorem is then used alongside Poljak's theorem to construct the various flows through the network and complete the argument. This discussion of Knight and Zelevinskii is the first and only in which a network flow argument is used for the implementation of either the Zelevinskii involution. In their paper, they do state that the link to the Mœglin-Waldspurger algorithm would be dealt with in a separate publication, however this paper doesn't appear to have materialised in any of their further work. Thus the following subsection will be devoted to studying the network, a slightly abridged version and their relation to the Mœglin-Waldspurger algorithm.

3.2.1 Network Theoretic Description

Firstly, we will present the network given by Knight and Zelevinskii in [9] as a single network. To do this we represent each integer contained inside of the multisegment as two vertices with an edge between of capacity of 1 to ensure that the integer is only used once, and take into account how the network changes for each pair of integers (i, j) contained in the multisegment. The formation of the network then follows:

Given a multisegment α with m segments, then let us arbitrarily index each segment in the multisegment α with an integer $1 \leq n \leq m$, thus $\alpha = \{\Delta_1, \ldots, \Delta_m\}$. Then for each integer i in the segment Δ, we will create two vertices $v_{i,\Delta,0}$ and $v_{i,\Delta,1}$ and add an edge from $v_{i,\Delta,0}$ to $v_{i,\Delta,1}$ with capacity 1. We will also create two extra vertices the source (s) and the sink (t). This network is currently highly disconnected so additional edges will be added in the following discussion - these edges will encode the preceding relations from the multisegment description. We will define an algorithm which completes the network with these edges and computes the associated dual multisegment.

Algorithm : Network Computation of the Dual

Following the precedes condition of the Mœglin-Waldspurger algorithm given in **Definition 2.5.3** we can see that given two segments in α

$$\Delta_1 = (b_1, \ldots, e_1) \text{ and } \Delta_2 = (b_2, \ldots, e_2),$$

where Δ_1 precedes Δ_2 then there are the following possible matchings from integers in Δ_2 to Δ_1 by the Mœglin-Waldspurger algorithm:

$$(\Delta_2, b_2) \to (\Delta_1, b_2 - 1),\ (\Delta_2, b_2 + 1) \to (\Delta_1, b_2),\ \cdots\cdots,\ (\Delta_2, e_1) \to (\Delta_1, e_1 - 1),\ (\Delta_2, e_1 + 1) \to (\Delta_1, e_1).$$

Hence for each integer n such that $b_2 \leq n \leq e_1 + 1$ there is a possible matching from n in Δ_2 to $n-1$ in Δ_1. To represent this in the network, we therefore construct an edge for each pair of segments Δ_1 and Δ_2 in which Δ_1 precedes Δ_2, and such n from $v_{n,\Delta_2,1}$ to $v_{n-1,\Delta_1,0}$ with capacity 1.

Fixing this network, we now run the following process for each ordered pair (i, j) such that $b_\alpha \leq i \leq j \leq e_\alpha$:

(i) We add an edge from the source (s) to $v_{j,\Delta,0}$ with capacity 1 for every Δ such that $j \in \Delta$.

(ii) Similarly, we add an edge from $v_{i,\Delta,1}$ to the sink (t) with capacity 1 for every Δ such that $i \in \Delta$.

(iii) Compute the maximum flow from the source to the sink through the corresponding network and denote this by $\overline{\alpha_{i,j}}$.

The values given by $\overline{\alpha_{i,j}}$ will be in the form of a triangle with the same dimensions as the original rank triangle for α.

Recall that the relations change with each iteration of the algorithm for both versions of the Mœglin-Waldspurger algorithm. We now want to prove that taking the maximum flow on this original network will simply correspond to carrying out the Mœglin-Waldspurger algorithm.

Theorem 3.2.2. *Given a multisegment* α *then*

$$\overline{\alpha_{i,j}} = \widetilde{\alpha_{i,j}},$$

where $\widetilde{\alpha_{i,j}}$ *denotes the* (i,j)*-th value of the rank triangle associated to the dual.*

Proof. This follows from the fact that the network presented in [9, Lemma 3.2] by Knight and Zelevinskii has equivalent flow at each pair (i,j) to our network, and they proved that this would be equal to the ranks of the dual in [9, Lemma 3.3] . □

Corollary 3.2.3. *The triangles of maximum flows* $\overline{\alpha_{i,j}}$ *have three key properties:*

1. $\overline{\alpha_{i,j}} \leq \overline{\alpha_{i,j-1}}$,

2. $\overline{\alpha_{i,j}} \leq \overline{\alpha_{i+1,j}}$, *and*

3. $\overline{\alpha_{l,k}} - \overline{\alpha_{l,j}} \leq \overline{\alpha_{i,k}} - \overline{\alpha_{i,j}}$, *where* $l < i, k \leq j$.

The fact that $\overline{\alpha_{i,j}}$ will produce an admissible triangle (**Proposition 2.2.2**) directly follows from **Theorem 3.2.2**. However we can also prove this using the argument of the independent flows:

Proof. Given an arbitrary multisegment α then the proofs of the three key properties using maximum flows are as follows:

1. Let assume that $\overline{\alpha_{i,j}} > \overline{\alpha_{i,j-1}}$, then by construction we have more flows from i to j than from i to $j-1$. This would contradict the fact that $\overline{\alpha_{i,j-1}}$ is maximum, since we could simply just end the flows from i to j at $j-1$ to increase $\overline{\alpha_{i,j-1}}$. Therefore $\overline{\alpha_{i,j}} \leq \overline{\alpha_{i,j-1}}$.

2. Let assume that $\overline{\alpha_{i,j}} > \overline{\alpha_{i+1,j}}$, then by construction we have more flows from i to j than from $i+1$ to j. This would contradict the fact that $\overline{\alpha_{i+1,j}}$ is maximum, since we could simply just start the flows from i to j at $i+1$ to increase $\overline{\alpha_{i+1,j}}$. Therefore $\overline{\alpha_{i,j}} \leq \overline{\alpha_{i+1,j}}$.

3. The expression given by $\overline{\alpha_{l,k}} - \overline{\alpha_{l,j}}$ provides the maximum number of flows that originating from l that can reach k but not j. Let us assume $\overline{\alpha_{l,k}} - \overline{\alpha_{l,j}} \leq \overline{\alpha_{i,k}} - \overline{\alpha_{i,j}}$, where $l < i, k \leq j$. Given any flow originating from l that can reach k but not j, then we can change the start point of this flow from l to i since $l < i$. Therefore we find that $\overline{\alpha_{l,k}} - \overline{\alpha_{l,j}} \leq \overline{\alpha_{i,k}} - \overline{\alpha_{i,j}}$, where $l < i, k \leq j$. □

3.2.2 Ford-Fulkerson Algorithm

The Ford-Fulkerson Algorithm can be used to find the maximum flow of a network, N. As we have already discussed the maximum flow through a network is required in the *Network Computation of the Dual*, and can therefore be used in generating the dual rank triangle by **Theorem 3.2.2**. The basic idea of the algorithm is to begin with an empty network and attempt to push additional flow from the source to the sink. This is carried out by searching for a so-called *augmenting paths* in which the flow can be increased along. If no augmenting path exists then the algorithm stops and the maximum flow has been achieved.

Note: It may be necessary to decrease the flow through a *backwards edge* to ultimately increase the flow through a network.

Definition 3.2.4 (Residual Network)**.** A residual network, R, can be described as $r : V \times V \to \mathbb{R}$ and is such that

$$r_{ij} = c_{ij} - f_{ij}.$$

The *residual network* contains edges where $r_{ij} \neq 0$. Therefore for the edges in the residual network we have two possibilities:

i) If $0 < r_{ij}$ and (i,j) is a forward edge then it is possible to increase the flow along the edge (i,j).

ii) If $0 < r_{ij}$ and (i,j) is a backward edge then it is possible to increase the backward flow along the edge (i,j) by decreasing the forward flow along the edge (j,i).

Definition 3.2.5 (Augmenting Path)**.** An augmenting path is a path $(i_1, i_2, \ldots, i_k)$ in the residual network, R, where $i_1 = s$, $i_k = t$ and $0 < r_{i_n i_{n+1}}$.

A network is at its maximum flow if and only if there is a feasible flow through the network and there exists no augmenting path in the residual network.

Example 3.2.6. Given the network, N, in Figure 3.4 with the flow, F, then the associated residual will be given by the residual network, R in Figure 3.5.

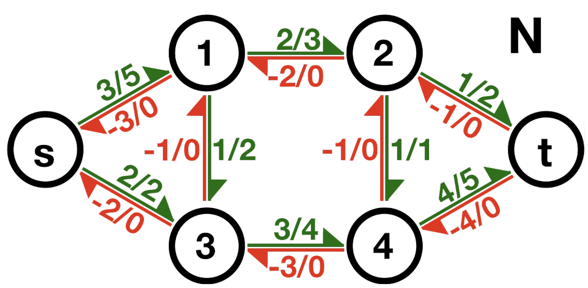


Figure 3.4: The original network, N, in which the flow and capacity are denoted by f/c. Note that the total flow from s to t is 5.

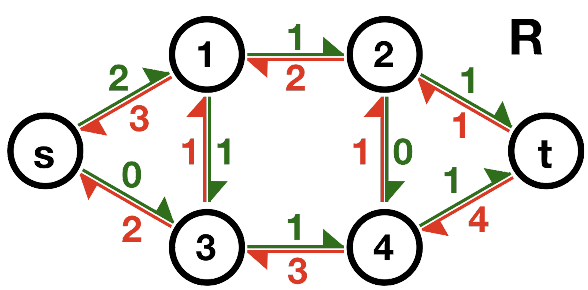


Figure 3.5: The respective residual network, R. Note that there is positive residual capacity on a number of paths eg. $p = \{s, 1, 2, t\}$, therefore p is an augmenting path of R.

The residual capacity of the path p is

$$r_{min} = min\{r_{s1}, r_{12}, r_{2t}\} = min\{2, 1, 1\} = 1.$$

For each augmenting path, the edge with the smallest residual capacity will be a bottleneck for the path. Hence the flow along the augmenting path can only be increased by a maximum of r_{min}.

Let us now formally define the Ford-Fulkerson algorithm.

Algorithm 1 Ford-Fulkerson

Begin
for all $(i,j) \in E$ **do** ▷ Sets initial flow to zero.
    $f_{ij} := 0$
    $f_{ji} := 0$
end for
while there is an augmenting path, $(i_1, i_2, \ldots, i_k)$, from s to t in N **do**
    $r_{min} := min\left(r_{i_1 i_2}, r_{i_2 i_3}, \ldots, r_{i_{k-1} i_k}\right)$
    Increase the flow along the augmenting path by r_{min}
end while
return f
End.

Example 3.2.7.

We will now use the Ford-Fulkerson algorithm to find the maximum flow in the network in Figure 3.6. We will arbitrarily pick the augmenting path since we have no pre-determined method for this. Firstly, we set the initial flow to be empty so the residual network will be the original network, N with 0 values for all backward flows. We will then use the augmenting paths seen right.

Note: Since $A \rightarrow D$ is a non-empty flow then we increase the negative flow $D \rightarrow A$ as stated in the algorithm in order to add our augmenting path ③ into the network.

Figure 3.6: A network, N.

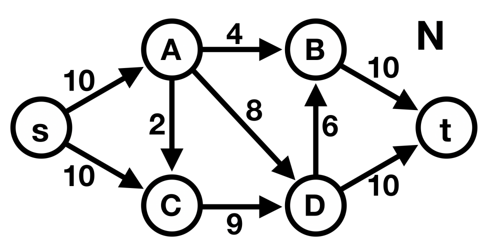


Table 3.2: Augmenting Paths

	Augmenting Path	r_{min}
① :	$s \rightarrow A \rightarrow D \rightarrow t$	8
② :	$s \rightarrow C \rightarrow D \rightarrow t$	2
③ :	$s \rightarrow C \rightarrow D \rightarrow A \rightarrow B \rightarrow t$	4
④ :	$s \rightarrow A \rightarrow D \rightarrow B \rightarrow t$	2
⑤ :	$s \rightarrow C \rightarrow D \rightarrow B \rightarrow t$	3

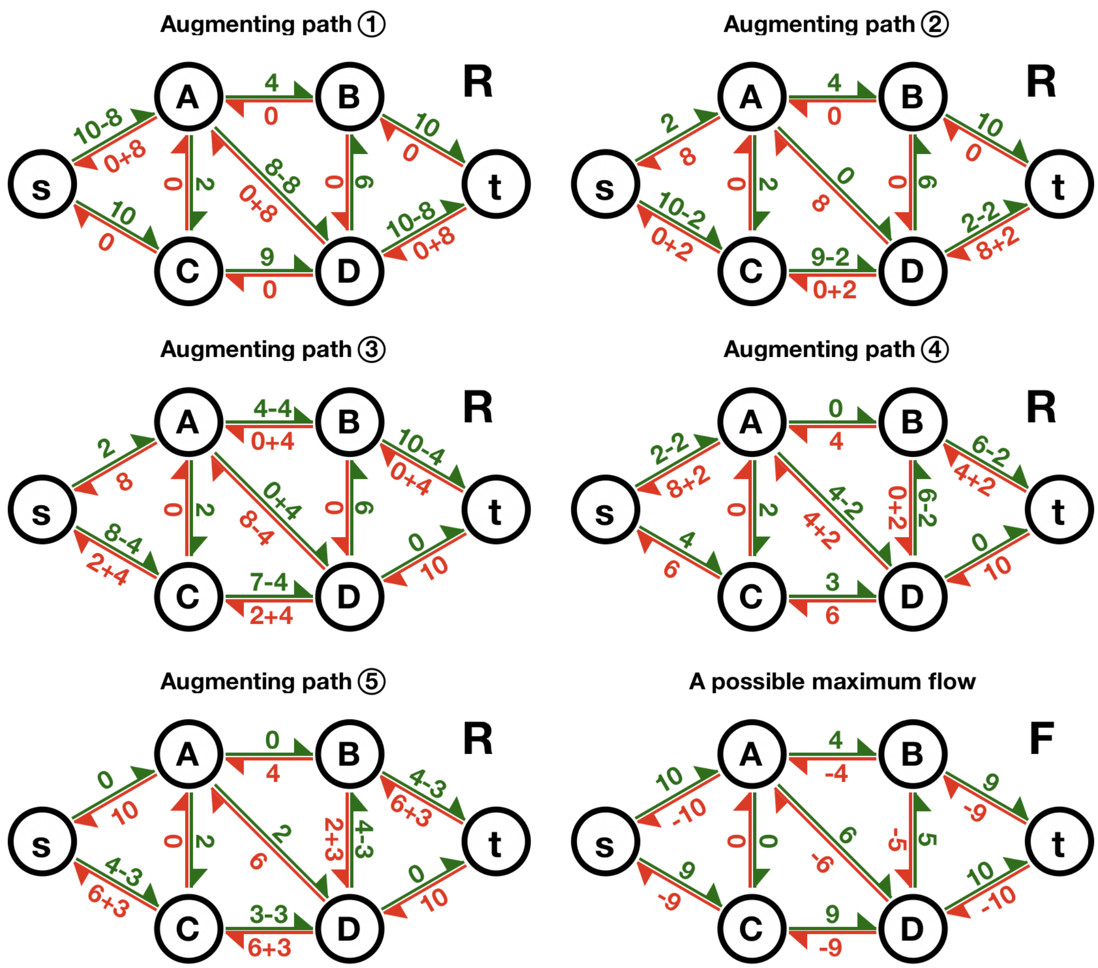


Figure 3.7: The iterations of the Ford-Fulkerson algorithm on the network.

Therefore we have found using the Ford-Fulkerson algorithm that the maximum flow through the network, N will be 19, which is found by taking the sum of the flows through each of the augmenting paths.

3.2.3 Relation to the Mœglin-Waldspurger Algorithm

The Mœglin-Waldspurger algorithm works by trying to find the maximum length chain from the maximum remaining value along the shortest preceding segments. We can seek to do something similar on our fixed network by pushing flows that will then correspond to the maximum flows of the network for each pair (i, j) when the sources and sinks are added. To do this, we must ensure that longer flows are always checked before shorter flows, so let us now denote an algorithm for implementing this.

Algorithm : Pushing Flows through Network

Let us denote the minimum and maximum integers in the multisegment α to be b_α and e_α respectively, then we will push the maximum flow along $[b,e]$ in the following order:

1. Set $e := e_\alpha$.

2. Set $b := b_\alpha$.

3. Push the maximum flow along $[b,e]$ by choosing edges associated the shortest segment which do not already have flow along them (if possible).

4. If $b \neq e$ then set $b := b+1$, and return to step 3.

5. If $b = e$ and $e > b_\alpha$, set $e := e-1$ and return to step 2.

Each value contained in the rank triangle must be finite, hence the multisegment α which is it associated to must contain a finite number of segments.

Lemma 3.2.8. *Taking the Pushing Flows through Network algorithm will still result in the elements being chosen in the correct order.*

In other words, given any segment $\Delta = [b,e]$ then for any integer n such that $b \leq n < e$, n must be chosen by the algorithm after $n+1$.

Proof. Let us assume that n from a segment $\Delta = [b,e]$ is the first such value from the multisegment α which has flow pushed through before before $n+1$ from Δ. Then there exists a segment $\Delta_1 = [b_1,e_1]$ that contains $n+1$ which precedes Δ, hence $b < b_1$, $e < e_1$, and $b_1 \leq e+1$. However, $n+2$ in Δ_1 must have had flow pushed through it before $n+1$, but $n+1$ in Δ did not have the same flow routed through it, since $n+1$ is chosen after n. There must exist a segment $\Delta_2 = [b_2,e_2]$ that is preceded by Δ_1, and is such that $n+2$ from Δ_1 is chosen to have the same flow pushed through it as $n+1$ from Δ_2. Note Δ_1 precedes both Δ, Δ_2 so $b_2 \geq b$ since the algorithm chooses the shorter segment first.

$(*)$ However, when $n+1$ is chosen from Δ_1, n from Δ_2 are not both chosen to have the same flow routed through them, then it results from the fact that n is chosen from Δ instead. Therefore either:

1. $n \notin \Delta_2$: So $b_2 = n+1$, but this would contradict the precedes condition since $b \leq n$ thus $b_2 \not< b$.

2. $n \in \Delta_2$: There exists $\Delta_3 = [b_3, e_3]$ containing $n+1$ that precedes Δ_2, and $n, n+1$ are chosen respectively from Δ_2, Δ_3 by the algorithm in the same dual segment.

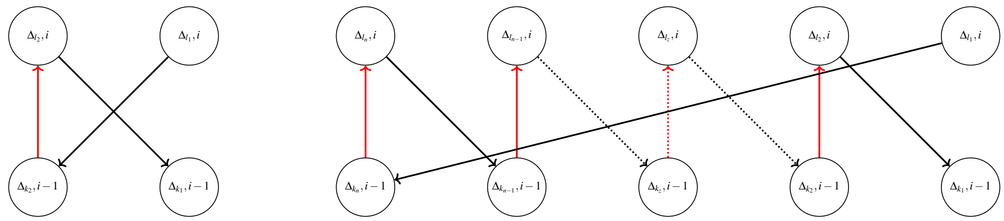
Since both $n, n+1 \in \Delta_2$ and Δ_3 precedes Δ_2 then $n+1 < e_3$. However $n+2$ must have been chosen from Δ_3 prior to $n+1$, but not in the same flow as $n+1$ from Δ_2 since this has already been chosen in a flow with $n+2$ from Δ_1 . Hence there exists a segment $\Delta_4 = [b_4, e_4]$ that precedes Δ_3. Note both Δ_2, Δ_4 precede Δ_3 so $b_4 \geq b_2$ since the algorithm chooses the shorter segment first. Thus we are back round to $(*)$ in our argument, so continuing this process recursively would result in a multisegment containing infinite segments, which contradicts the formation of G. Therefore there can never be an $n, n+1$ in a segment of α in which n is chosen before $n+1$ by the algorithm. □

Note that the flow $[b, e]$ will be completely contained in the flows $[b-n, e+m]$ for $0 \leq n \leq b - b_\alpha$ and $0 \leq m \leq e_\alpha - e$, however the maximum flows for each $[b-n, e+m]$ will have already been pushed before $[b, e]$.

Lemma 3.2.9. *If we use the Pushing Flows through Network Algorithm, then we will achieve the maximum flow for each pair (b, e).*

Proof. The Ford-Fulkerson algorithm works by simply pushing flows through the network. If we begin at $[b_\alpha, e_\alpha]$ then we can simply push the maximum flow. We can then set this to be a flow through the network. Following this, we can then move on to the next iteration $[b_\alpha + 1, e_\alpha]$ (with sources and sinks connected to the vertices associated 0 for $b_\alpha + 1$ and 1 for e_α respectively). The order of the process ensures that all flows containing $[b_\alpha + 1, e_\alpha]$ have already been chosen. Therefore on this iteration it will be the last one for which additional flow can be pushed from $[b_\alpha + 1, e_\alpha]$. We already have a feasible flow given by the previous iteration(s), and thus the Ford-Fulkerson algorithm states that in order to achieve the maximum flow we must push any augmenting flows.

Let us assume that we are required to use a backwards edge from i to $i+1$ in order to increase the flow, hence decrease an already allocated flow along the edge from $i+1$ to i. Given that the network has a very rigid structure then we have a very restrictive formation for the flow between i and $i+1$ that use backwards edges.



(a) Smallest Backwards Flow (b) Generalised Backwards Flow

Figure 3.8: Possible backwards flows for each pair (i, j).

The edges in each graph represent preceding relations. If we denote $[b_i, e_i]$ to be the original segment Δ_i. Then we can immediately say that

$$b_{l_1} \leq i = (i-1) + 1 \leq e_{k_1} + 1.$$

Likewise we also have that $b_{k_n} < b_{l_1}$, and given that the shortest segment is always chosen then $b_{k_j} \leq b_{k_{j+1}}$. Therefore we find that

$$b_{k_1} \leq b_{k_2} \leq \cdots \leq b_{k_n} < b_{l_1}.$$

By **Lemma 3.2.8**, a from a segment Δ must always be chosen for a flow before $a-1$. If we assume that $i \in \Delta_{k_1}$, then it must have been chosen in a previous flow with $i+1$ from a segment Δ or to begin a new flow. If we assume that $e_{k_1} \geq e_{l_1}$, then this implies that if Δ exists then Δ_{l_1} would also be preceded by Δ. However if this was the case then i from Δ_{l_1} should have been chosen instead of i from Δ_{k_1} since it is shorter. Therefore we find that $e_{k_1} < e_{l_1}$ when $i \in \Delta_{k_1}$. Otherwise, let us assume $i \notin \Delta_{k_1}$ then $e_{k_1} = i-1$, hence

$$e_{k_1} = i - 1 < i \leq e_{l_1}.$$

Overall we have found that we have the following inequalities relating Δ_{l_1} and Δ_{k_1}: $b_{l_1} \leq e_{k_1} + 1$, $b_{k_1} < b_{l_1}$, and $e_{k_1} < e_{l_1}$, which establishes that Δ_{l_1} originally precedes Δ_{k_1} and hence there exists an edge between Δ_{l_1}, i and $\Delta_{k_1}, i-1$ in our network. This edge should instead be used in place of the flow which uses backwards edges.

We have therefore demonstrated that a backwards edge will never be required when choosing the shortest possible segment each time, thus we can push each of these augmented flows through the original network without changing any of the previous maximum flows. We can recursively continue this argument made for $[b_\alpha + 1, e_\alpha]$ for all $[b,e]$ in the order described in the algorithm to then find that the final flow pushed through the network will achieve the maximum flow for each pair (b,e). $\square$

Remark 3.2.10. A natural question to ask is: how do these individual flows which are passed through the network by algorithm correspond to the multisegment description? If we look at an individual flow from e to b which is pushed through the network at an iteration, then it corresponds to a new augmenting flow which wasn't previously pushed by a longer flow containing $[b,e]$. If we now look at the multiplicity of these new flows for each $[b,e]$ and denote it by $F_{b,e}$. Then we find

$$\begin{aligned} F_{b,e} &= (\text{Maximum Flow from } e \text{ to } b) - (\text{Maximum Flow of longer flows containing } e \text{ to } b), \\ &= \overline{\alpha_{b,e}} - (\overline{\alpha_{b-1,e}} + \overline{\alpha_{b,e+1}} - \overline{\alpha_{b-1,e+1}}), \\ &= \widetilde{\alpha_{b,e}} - \widetilde{\alpha_{b-1,e}} - \widetilde{\alpha_{b,e+1}} + \widetilde{\alpha_{b-1,e+1}}, \\ &= \widetilde{m_{b,e}}. \end{aligned}$$

Therefore we have found that the multiplicity of these new flows is equivalent to the multiplicity of the segments in the dual multisegment for each (b,e).

Theorem 3.2.11. *The Pushing Flows through Network algorithm will generate flows which will each correspond to a segment in the dual multisegment.*

Proof. This follows from **Lemma 3.2.9** that states that the algorithm will achieve a flow that is equal to all of the maximum ranks, and **Remark 3.2.10** that shows that these flows will then be equivalent to the multisegments from the rank triangle. $\square$

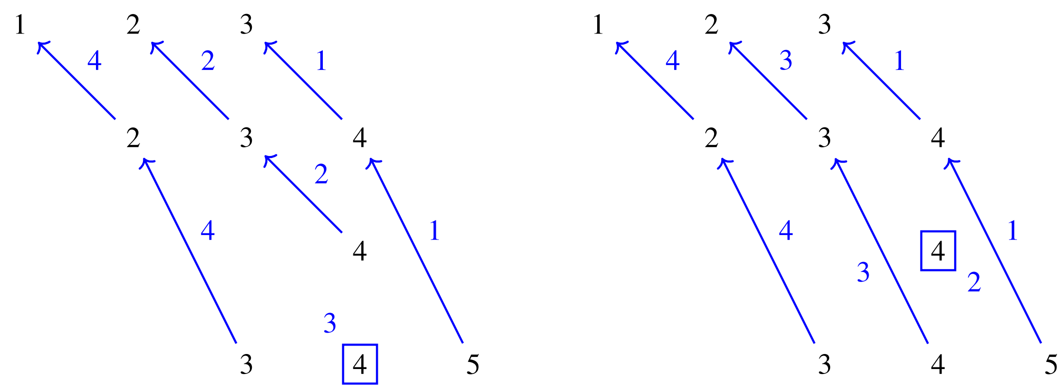
Corollary 3.2.12. *The Mœglin-Waldspurger algorithm can be carried out using only the original preceding relations.*

In other words at steps 2 and 3 in the Mœglin-Waldspurger algorithm we can change the precedes condition on Δ_m to be that Δ_{m-1} must have originally preceded.

We have therefore found that a method for pushing maximum flows can be used with the implicit choice that we choosing the shortest segment each time.

We will illustrate this through the following example:

Example 3.2.13. Let $\alpha = \{[1,3],[2,4],[3,5],[4]\}$, then we can now run both the original Mœglin-Waldspurger algorithm and the abridged version, which only uses the initial preceding relations.



(a) Original Algorithm (b) Abridged Algorithm

Figure 3.9: The implementation both versions of the Mœglin-Waldspurger algorithm.

As expected, both versions of the Mœglin-Waldspurger algorithm find the same dual of multi-segment for α, which is $\tilde{\alpha} = \{[1,3],[2,4],[3,5],[4]\}$. Notice that a precedes relation is generated following the first iteration, and is then used by the original algorithm. We can also see from this example that the abridged version will not always find the segments in the same order as the original. This follows from the fact that it does not take advantage of use of the new preceding relations which are generated with each iteration, and hence will not always be able to choose the longest paths in descending order for each end value. It therefore follows that **Corollary 3.1.16** will not be true for this abridged version of the algorithm.

In this, section we have discussed two significant combinatorial interpretations of the Zelevinskii involution. The first, interpretation is that for pairs (i,j) the maximum flow through the network of the initial preceding relations will be equal to the ranks computed by the Mœglin-Waldspurger algorithm. The second is that exists an abridged version of the Mœglin-Waldspurger algorithm, which allows us to find the dual multisegment by fixing the initial preceding relations. These two interpretations will be of massive importance in Chapter 4 as we study some of the conjectures in the local Langlands correspondence which were first discussed in Chapter 2.

3.2.4 Example of Network Computation of the Dual

We will now use the *Network Computation of the Dual* algorithm on the same example detailed in **Example 3.1.2** to demonstrate that this new algorithm will also compute the associated dual. Let us consider the multisegment α, and label the segments contained inside it as follows:

$$\alpha = \{[1],[1,2],[2,3],[2,3],[3],[3,4]\} = \{\Delta_1,\Delta_2,\Delta_3,\Delta_4,\Delta_5,\Delta_6\}.$$

We can now create the fixed network which we will run the algorithm on. This network is shown in **Figure 3.10**, and outlines the preceding relations between segments in α.

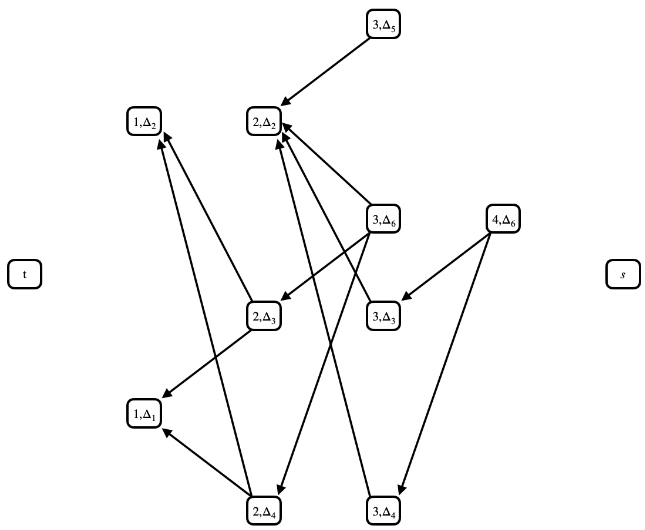


Figure 3.10: The fixed network of preceding relations.

Note that each edge in the network has capacity 1 and to simplify the diagram we have collapsed the two vertices $v_{n,\Delta_i,0}$ and $v_{n,\Delta_i,1}$ onto a single vertex indexed by n,Δ_i, thus each vertex in the network (except for the source and sink) will have a restriction that only a single flow can be sent through it.

We can now run the algorithm for all ordered pairs (i,j) such that $1 \leq i \leq j \leq 4$. If we look at the case in which $(i,j) = (1,4)$, then we construct edges from the source to all vertices with initial index 4 and from all vertices with initial index 1. The corresponding network is shown in **Figure 3.11**. If we then implement the Ford-Fulkerson algorithm on this network, then we find the maximum flow to be 0 which thus implies $r_{1,4} = 0$.

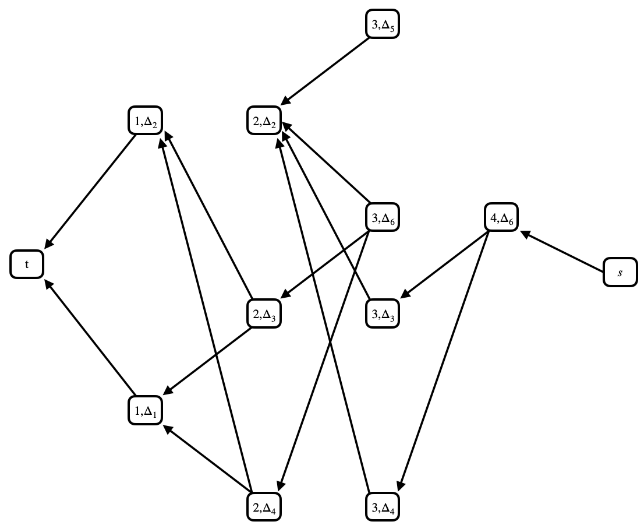


Figure 3.11: The network constructed for $(i, j) = (1,4)$.

Continuing this process for all pairs (i, j), we obtain the following rank triangle for the dual

$$\begin{array}{ccccccc} \widetilde{\alpha_{1,1}} & & \widetilde{\alpha_{2,2}} & & \widetilde{\alpha_{3,3}} & & \widetilde{\alpha_{4,4}} \\ & \widetilde{\alpha_{1,2}} & & \widetilde{\alpha_{2,3}} & & \widetilde{\alpha_{3,4}} & \\ & & \widetilde{\alpha_{1,3}} & & \widetilde{\alpha_{2,4}} & & \\ & & & \widetilde{\alpha_{1,4}} & & & \end{array} = \begin{array}{ccccccc} 2 & & 3 & & 4 & & 1 \\ & 2 & & 2 & & 1 & \\ & & 1 & & 1 & & \\ & & & 0 & & & \end{array}.$$

This rank triangle above corresponds to the following multisegment

$$\tilde{\alpha} = \{[1,2],[1,3],[3],[3],[2,4]\},$$

which will be the multisegment associated to the dual. Therefore the *Network Computation of the Dual* algorithm gives the same dual as the two algorithms shown in the example in **Section 3.1**.

3.3 Complete Example

Now that we have established both the interpretation of objects in GL_n and a method for computing the dual orbit in Chapter 2, then it makes sense to present a complete example illustrating them. Let us fix $G = GL_{16}(\mathbb{C})$, if we study the Vogan variety determinded by

$$\lambda = \mathrm{diag}(q^2, q^2, q, q, q, q, 1, 1, 1, 1, q^{-1}, q^{-1}, q^{-1}, q^{-1}, q^{-2}, q^{-2}),$$

then $V_\lambda \subseteq \mathfrak{gl}_{16}(\mathbb{C})$. We can index these eigenvalues as follows:

$$\lambda_0 = q^{-2},\ \lambda_1 = q^{-1},\ \lambda_2 = 1,\ \lambda_3 = q,\ \lambda_4 = q^2.$$

Therefore V_λ will be a matrix of the following form:

$$V_\lambda = \left\{ \begin{pmatrix} 0 & x_4 & 0 & 0 & 0 \\ 0 & 0 & x_3 & 0 & 0 \\ 0 & 0 & 0 & x_2 & 0 \\ 0 & 0 & 0 & 0 & x_1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \middle| \begin{array}{c} x_4 \in \mathrm{Mat}_{2,4}(\mathbb{C}) \\ x_2, x_3 \in \mathrm{Mat}_{4,4}(\mathbb{C}) \\ x_1 \in \mathrm{Mat}_{4,2}(\mathbb{C}) \end{array} \right\}.$$

Let us denote E_{λ_i} to be the eigenspace of $\lambda(\mathrm{Frob})$ with eigenvector λ_i, then

$$V_\lambda = \mathrm{Hom}(E_{\lambda_3}, E_{\lambda_4}) \times \mathrm{Hom}(E_{\lambda_2}, E_{\lambda_3}) \times \mathrm{Hom}(E_{\lambda_1}, E_{\lambda_2}) \times \mathrm{Hom}(E_{\lambda_0}, E_{\lambda_1}),$$

so V_λ is a representation variety for the quiver of type A:

$$\lambda_0 \xrightarrow{\quad x_1 \quad} \lambda_1 \xrightarrow{\quad x_2 \quad} \lambda_2 \xrightarrow{\quad x_3 \quad} \lambda_3 \xrightarrow{\quad x_4 \quad} \lambda_4 \ .$$

Thus we can represent $x \in V_\lambda$ as follows:

$$x = (x_4, x_3, x_2, x_1) \in \mathrm{Mat}_{2,4}(\mathbb{C}) \times \mathrm{Mat}_{4,4}(\mathbb{C}) \times \mathrm{Mat}_{4,4}(\mathbb{C}) \times \mathrm{Mat}_{4,2}(\mathbb{C}).$$

Similarly, $H_\lambda = \mathrm{Aut}(E_{\lambda_4}) \times \mathrm{Aut}(E_{\lambda_3}) \times \mathrm{Aut}(E_{\lambda_2}) \times \mathrm{Aut}(E_{\lambda_1}) \times \mathrm{Aut}(E_{\lambda_0})$, so

$$h = (h_4, h_3, h_2, h_1, h_0) \in GL_2(\mathbb{C}) \times GL_4(\mathbb{C}) \times GL_4(\mathbb{C})) \times GL_4(\mathbb{C}) \times GL_2(\mathbb{C}).$$

Thus the action of $H_\lambda \times V_\lambda \to V_\lambda$ is given by

$$(h_4,h_3,h_2,h_1,h_0)\cdot(x_4,x_3,x_2,x_1) := \left(h_4x_4h_3^{-1},h_3x_4h_2^{-1},h_2x_4h_1^{-1},h_1x_4h_0^{-1}\right).$$

If we fix the matrix representation V_λ, then we can study the orbit and also compute the associated dual, so let

$$x_4 = \begin{bmatrix}1&0&0&0\\0&0&1&0\end{bmatrix}, x_3 = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}, x_2 = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}, x_1 = \begin{bmatrix}0&0\\1&0\\0&0\\0&1\end{bmatrix}.$$

Then we can compute the rank triangle as follows:

2 4 4 4 2	(Given by the multiplicities of the eigenvalues)
2 3 3 2	(Ranks of the initial matrices x_1,x_2,x_3,x_4)
1 2 1	(Ranks of the compositions x_2x_1,x_3x_2,x_4x_3)
1 1	(Ranks of the compositions $x_3x_2x_1,x_4x_3x_2$)
0	(Ranks of the composition $x_4x_3x_2x_1$)

We can then use **Proposition 2.3.4** in order to construct the triangle for the multiplicity of segments in the associated multisegment

$$\begin{array}{ccccccccc}0&&0&&0&&0&&0\\&1&&1&&1&&1&\\&&0&&0&&0&&\\&&&1&&1&&&\\&&&&0&&&&\end{array}$$

This corresponds to the multisegment

$$\alpha = \{[1,2],[2,3],[3,4],[4,5],[1,4],[2,5]\} = \{\Delta_1,\Delta_2,\Delta_3,\Delta_4,\Delta_5,\Delta_6\}.$$

We can now create the fixed network which we will run the algorithm on. This network is shown in **Figure 3.12**, and outlines the preceding relations between segments in α.

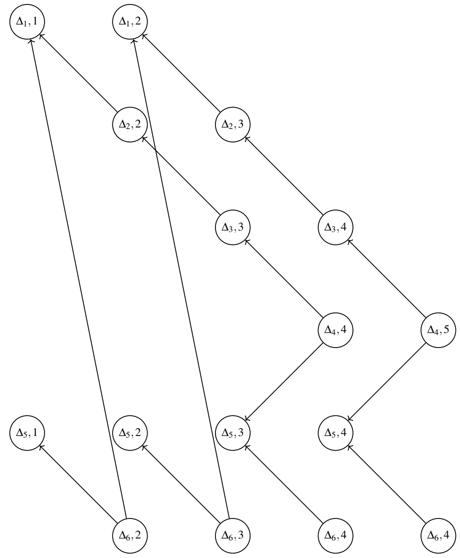


Figure 3.12: The fixed network of preceding relations for the complete example.

Note that each edge in the network has capacity 1 and to simplify the diagram we have collapsed the two vertices $v_{n,\Delta_i,0}$ and $v_{n,\Delta_i,1}$ onto a single vertex indexed by n, Δ_i, thus each vertex in the network (except for the source and sink) will have a restriction that only a single flow can be sent through it.

We can now run the algorithm for all ordered pairs (i, j) such that $1 \leq i \leq j \leq 5$. Following this process, we obtain the following rank triangle for the dual

$$\begin{array}{ccccccccc} 2 & & 4 & & 4 & & 4 & & 2 \\ & 2 & & 3 & & 3 & & 2 & \\ & & 1 & & 2 & & 1 & & \\ & & & 1 & & 1 & & & \\ & & & & 0 & & & & \end{array}$$

This rank triangle above corresponds to the following multisegment

$$\tilde{\alpha} = \{[1,2],[2,3],[3,4],[4,5],[1,4],[2,5]\},$$

which is equal to the original multisegment so α is *self-dual* (**Definition 3.1.3**).

Chapter 4 : Combinatorics of Numerical Invariants

We have previously established a number of different methods for finding the dual of a multisegment, however we are yet to study properties and relations satisfied by the multisegments corresponding to specific families of ABV-packets. Thus we now devote this chapter to further our investigation into ABV-packets, and will use the previously discussed methods and some new combinatorial approaches to do this. We will first seek to study in **Section 4.1** the effect of the Mœglin-Waldspurger algorithm when we partition a multisegments into sub-multisegments and then run the algorithm on each of the individual partitions. Following this in **Section 4.2**, we will then examine multisegments in terms of different numerical invariants and characteristics. Through which we will prove that a large collection of α's will satisfy the *partial ordering relation*:

$$\text{For all multisegments } \beta \text{ such that } \alpha \leq \beta \text{ and } \tilde{\alpha} \leq \tilde{\beta}, \text{ we find } \alpha = \beta.$$

This inspection will lead us to examining **Remark 2.4.6** and the **Open Orbit Conjecture 2.4.7**.

These families of α's for which the partial ordering relation will be satisfied are outlined in **Theorem 4.2.9**, **Theorem 4.2.23**, **Theorem 4.2.35**, and **Theorem 4.2.39**. The work on *simple multisegments* and *ladder multisegments*, in **Theorem 4.2.9** and **Theorem 4.2.23** respectively, is based upon unpublished work by my supervisor Dr. Fiori, however our main focus will be my generalisation of these results to much larger and more significant families of multisegments. Most significantly in **Section 4.2**, the numerical invariants will be used to prove the important conjecture:

ABV-packets for orbits of Arthur type in GL_n are singletons,

by means of **Theorem 4.2.35**. Finally, we will flip the problem on its head and present a method for generating multisegments which violate the partial ordering relation in **Section 4.3**, and present a conjecture with subsequent corollaries following a numerical study.

4.1 Endoscopic Decomposition

For any multisegment α, we can decompose it into sub-multisegments α_i's such that

$$\alpha = \bigsqcup_i \alpha_i.$$

On the level of conjugacy classes this corresponds to a decomposition of each of the vector spaces

$$V_k = \bigoplus_i V_k^{(i)}$$

into a direct sum. One can then consider the quiver representations which preserve this decomposition. Such a decomposition corresponds to an inclusion of an *endoscopic subgroup*[4].

The requirement that $\alpha = \bigsqcup_i \alpha_i$ implies that there exists a collection of quiver representations for each of the α_i's that combine to form one of type α. It is automatic that any element of the conormal bundle arising in the smaller endoscopic subgroup induces one for the larger group. That is, the union

$$\bigsqcup_i \widetilde{\alpha}_i$$

will give us elements which commute with representations for α.

The total number of possible endoscopic decompositions is equal the total number of possible partitions of a set. The sequence of numbers describing the number of partitions of a set are eponymously named *Bell numbers* after Eric Temple Bell, who wrote about them in his work on Bell polynomials in 1934 [3]. The Bell numbers are denoted by B_n and the sequence which follows starts at $n = 0$:

$$1, 1, 2, 5, 15, 52, 203, 877, 4140, \ldots;^5$$

where each B_n will correspond to the number of possible endoscopic decompositions of a multisegment that contains n segments.

[4]This terminology arises from the connection to endoscopy in the Langlands program, however we will solely focus on the relation to the multisegment description in this thesis.

[5]The sequence of Bell numbers is given by A000110 from the On-Line Encyclopaedia of Integer Sequences.

Given the increasingly large number of possible decompositions, it may come as no surprise that if a decomposition of α_i's is chosen at random, then it will typically not be such that

$$\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}.$$

However, we will certainly have

$$\tilde{\alpha} \geq \bigsqcup_i \tilde{\alpha}_i.$$

Example 4.1.1. Let us consider the multisegment

$$\alpha = \{(123), (234), (345)\},$$

then according to the Bell numbers there will be five possible endoscopic decompositions. For each of these decompositions, we will study $\sqcup_i \widetilde{\alpha_i}$ and $\widetilde{\sqcup_i \widetilde{\alpha_i}}$.

1. The *trivial decomposition* in which α remains fixed, so $\alpha_1 = \alpha$. Note α is self dual ($\alpha = \tilde{\alpha}$). Hence $\sqcup_{i=1} \widetilde{\alpha_i} = \tilde{\alpha} = \alpha$ and $\widetilde{\sqcup_{i=1} \widetilde{\alpha_i}} = \tilde{\tilde{\alpha}} = \alpha$.

2. The *complete decomposition* in which $\alpha_1 = \{(123)\}$, $\alpha_2 = \{(234)\}$ and $\alpha_3 = \{(345)\}$. Then

$$\bigsqcup_{i=1,2,3} \widetilde{\alpha_i} = \{(1), (2), (2), (3), (3), (3), (4), (4), (5)\},$$

and

$$\widetilde{\bigsqcup_{i=1,2,3} \widetilde{\alpha_i}} = \{(12345), (234), (3)\},$$

which is formed by the union intersection of (123) with (345) in α.

3. The endoscopic decomposition in which $\alpha_1 = \{(123), (234)\}$ and $\alpha_2 = \{(345)\}$. Then

$$\bigsqcup_{i=1,2} \widetilde{\alpha_i} = \{(12), (23), (3), (34), (4), (5)\},$$

and

$$\widetilde{\bigsqcup_{i=1,2} \widetilde{\alpha_i}} = \{(123), (34), (2345)\},$$

which is formed by the union intersection of (234) with (345) in α.

4. The endoscopic decomposition in which $\alpha_1 = \{(123)\}$ and $\alpha_2 = \{(234),(345)\}$. Then

$$\bigsqcup_{i=1,2} \widetilde{\alpha}_i = \{(1),(2),(23),(3),(34),(45)\},$$

and

$$\widetilde{\bigsqcup_{i=1,2} \widetilde{\alpha}_i} = \{(1234),(23),(345)\},$$

which is formed by the union intersection of (123) with (234) in α.

5. The endoscopic decomposition in which $\alpha_1 = \{(123),(345)\}$ and $\alpha_2 = \{(234)\}$. Then

$$\bigsqcup_{i=1,2} \widetilde{\alpha}_i = \{(1),(2),(23),(3),(34),(4),(5)\},$$

and

$$\widetilde{\bigsqcup_{i=1,2} \widetilde{\alpha}_i} = \{(12345),(23),(34)\},$$

which is formed by the union intersection of (123) with (234) in α followed by (1234) with (345).

As seen in each of the decompositions we can form $\widetilde{\sqcup_i \widetilde{\alpha}_i}$ using union intersection on α. Thus we now seek to generalise this idea and prove a number of key facts about endoscopic decomposition. Let us now define n_α to be equal to the number of segments in α. Note that we will further discuss a number of other numerical invariants in **Section 4.2**.

An alternate way of looking at endoscopic decomposition is that it splits up the network discussed in **Section 3.2** into a collection of subnetworks, which we will then compute the *Network Computation of the Dual* algorithm on in order to construct $\bigsqcup_i \widetilde{\alpha}_i$. We will use this alternate description in the proof of the following:

Lemma 4.1.2. *Let $\alpha = \bigsqcup_i \alpha_i$ be any endoscopic decomposition of α, then*

$$n_{\widetilde{\alpha}} \leq n_{\bigsqcup_i \widetilde{\alpha}_i} = \sum_i n_{\widetilde{\alpha}_i} \leq N,$$

where N denotes the total number of integers (including their repeated appearances) in the multi-segment α.

Proof. Firstly, there can never be a case in which an empty segment is contained in the multisegment. Since the Mœglin-Waldspurger algorithm preserves the value N number of total elements in the dual multisegment, then the maximum number of segments in $\bigsqcup_i \widetilde{\alpha}_i$ will occur when every segment has a single element. This occurs when the α_i's are equal to the individual elements of α, since each $\widetilde{\alpha}_i$ will then be made up of all singletons, and thus the overall union $\bigsqcup_i \widetilde{\alpha}_i$ will simply consist of N singletons.

We also have the trivial decomposition in which the multisegment α remains fixed, since it is broken into one partition. In this case $\bigsqcup_i \widetilde{\alpha}_i = \tilde{\alpha}$, so there will be $n_{\tilde{\alpha}}$. Let us assume that there exists a decomposition such that $\sum_i n_{\widetilde{\alpha}_i} < n_{\tilde{\alpha}}$. If we use the network description from Chapter 3 then an endoscopic decomposition forms subnetworks from a larger network and hence edges are removed since the initial preceding relations will only remain inside each of these sub-multisegments in the decomposition. This will result in the overall sum of the flows at each iterative step will being less than or equal to the original. Therefore, there can never exist a decomposition such that $\sum_i n_{\widetilde{\alpha}_i} < n_{\tilde{\alpha}}$, since this would require at least one segment to be of greater length. So, we have obtained the required inequality

$$n_{\tilde{\alpha}} \leq \sum_i n_{\widetilde{\alpha}_i} \leq N.$$

□

We can also study how an endoscopic decompositions of multisegment will relate to the original multisegment in terms of a partial ordering.

Proposition 4.1.3. *Given an endoscopic decomposition $\alpha = \bigsqcup_i \alpha_i$, then either:*

1. $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha}_i$,
2. *or, $\tilde{\alpha}$ can be recovered by taking a combination of the union intersection and conjunction actions on $\bigsqcup_i \widetilde{\alpha}_i$, that is, $\bigsqcup_i \widetilde{\alpha}_i \leq \tilde{\alpha}$.*

Proof. Let $\alpha = \bigsqcup_i \alpha_i$ be an endoscopic decomposition, then:

1. $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha}_i$, will be the trivial case.
2. Alternatively, let $\tilde{\alpha} \neq \bigsqcup_i \widetilde{\alpha}_i$, and C, D be the rank triangles associated to $\bigsqcup_i \widetilde{\alpha}_i, \tilde{\alpha}$. Then if we use the network theoretic approach from Chapter 3 then C is formed from subnetworks

of the network D. Therefore the overall sum of the flow associated to C will be less than or equal to the flow through D for each (i,j), that is, $c_{i,j} \leq d_{i,j}$ for all pairs (i,j). Also note that $c_{i,i} = d_{i,i}$ for all i, since the associated vertices will remain the same hence $C \leq D$, and by **Proposition 2.5.6** we have $\bigsqcup_i \widetilde{\alpha}_i \leq \tilde{\alpha}$. Therefore $\tilde{\alpha}$ can be recovered by taking a combination of the union intersection and conjunction actions on $\bigsqcup_i \widetilde{\alpha}_i$. □

Proposition 4.1.4. *Suppose that α is any multisegment with corresponding conjugacy class C. Also let $\bigsqcup_i \alpha_i$ be an endoscopic decomposition of α, and denote $\tilde{D}$ to be the conjugacy class associated to $\bigsqcup_i \widetilde{\alpha}_i$. Thus $D = \tilde{\tilde{D}}$ will be the conjugacy class associated to $\widetilde{\bigsqcup_i \widetilde{\alpha}_i}$. Then*

$$C \leq D \ \textit{ and } \ \tilde{D} \leq \tilde{C}.$$

Proof. Firstly, if $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha}_i$ then it immediately follows that $\tilde{D} = \tilde{C}$. As a consequence, we will also have that $C = D$.

Alternatively, if $\tilde{\alpha} \neq \bigsqcup_i \widetilde{\alpha}_i$ then we use **Proposition 4.1.3** to conclude that $\bigsqcup_i \widetilde{\alpha}_i \leq \tilde{\alpha}$. We can then relate the multisegment partial ordering to the partial ordering of ranks by using **Proposition 2.5.5** to find that

$$\tilde{D} \leq \tilde{C}.$$

What remains is to prove $C \leq D$ or equivalently $\alpha \leq \widetilde{\bigsqcup_i \widetilde{\alpha}_i}$. To do this we first recall that $\alpha_i = \tilde{\tilde{\alpha}}_i$, and hence $\alpha = \bigsqcup_i \tilde{\tilde{\alpha}}_i$. If we set $\tilde{\beta} = \bigsqcup_i \widetilde{\alpha}_i$ then by **Proposition 4.1.3**

$$\begin{aligned} \bigsqcup_i \tilde{\tilde{\alpha}}_i &\leq \tilde{\tilde{\beta}}, \\ \alpha &\leq \widetilde{\bigsqcup_i \widetilde{\alpha}_i}, \end{aligned}$$

which implies by **Proposition 2.5.5** that

$$C \leq D.$$ □

Therefore we have proved that the relation shown in **Example 4.1.1** will hold for all endoscopic decompositions. Given the increasingly large number of possible endoscopic decompositions, then it will often therefore be the case that when $C < D$ we will also have $\tilde{D} < \tilde{C}$.

4.2 Combinatorics of Numerical Invariants

Given any multisegment α then we will use the following six numerical invariants to study them:

i) $e_\alpha := \max(\alpha)$;

ii) $L_\alpha :=$ Length of the longest segment in α;

iii) $n_\alpha :=$ Number of segments in α;

iv) $c_\alpha :=$ Minimum number of segments in which $\cup_{\Delta\in\alpha}\Delta$ constructs;

Let us denote the segments generated by the minimal formation of $\cup_{\Delta\in\alpha}\Delta$ to be $\Delta^1,\dots,\Delta^{c_\alpha}$.

v) $S_\alpha := \sum_{i=1}^{c_\alpha} \left|\Delta^i\right|$;

vi) $C_\alpha :=$ Maximum number of components in a decomposition $\alpha = \bigsqcup_i \alpha_i$ for which $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}$.

Note that if $c_\alpha = 1$, then there exists a single segment in the minimal formation of $\cup_{\Delta\in\alpha}\Delta$. In this case, let us define $S_\alpha := |\cup_{\Delta\in\alpha}\Delta|$.

Example 4.2.1. Given a multisegment

$$\alpha = \{[0,1],[1,3],[2,2],[3,4]\},$$

then we can categorise it as follows.

i) $e_\alpha = \max(\alpha) = 4$;

ii) $L_\alpha =$ Length of $[1,3]$ which is the longest segment in $\alpha = 3$;

iii) $n_\alpha =$ Number of segments in $\alpha = 4$;

iv) $c_\alpha := \cup_{\Delta\in\alpha}\Delta = [0,4]$ so the minimum number of segments in which it constructs $= 1$;

v) $S_\alpha := |\cup_{\Delta\in\alpha}\Delta| = |[0,4]| = 5$;

vi) $C_\alpha :=$ Maximum number of components in a decomposition $\alpha = \bigsqcup_i \alpha_i$ for which $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}$.

There exists a decomposition $\alpha_1 = \{[0,1],[1,3],[3,4]\}$ and $\alpha_2 = \{[2,2]\}$, which satisfies the property so C_α is at least two. Note that α is self dual so there exists a segment of length 3 in the dual, which can only result from a component that contains three or more segments. However by the pigeonhole principle, if there are at least three non-empty components in the decomposition then a component which contains three segments cannot exist. Thus $\tilde{\alpha} \neq \bigsqcup_i \widetilde{\alpha_i}$, when $C_\alpha \geq 3$ since the segment of length 3 in the dual cannot be generated by the Mœglin-Waldspurger algorithm on the individual components. Therefore C_α is less than three, so we can conclude $C_\alpha = 2$.

Note that studying the case in which $c_\alpha > 1$ is unnecessary as there is effectively no interaction between the individual components, so we could simply consider them individually. Thus we will call a multisegment α *connected* if $c_\alpha = 1$. Further, we will call a multisegment α *irreducible* when $C_\alpha = 1$ and hence the decomposition $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}$ into C_α number of components will be the *irreducible decomposition*.

Lemma 4.2.2. *Let α, β be multisegments and $\tilde{\alpha}, \tilde{\beta}$ their respective dual multisegments. Then*

i) For any two multisegments α, β with isomorphic quiver representations, $c_\alpha = c_\beta$ and $S_\alpha = S_\beta$.

ii) If $\alpha \leq \beta$ then $L_\alpha \leq L_\beta$.

iii) If $\alpha \leq \beta$ then $n_\alpha \geq n_\beta$.

iv) $n_{\tilde{\alpha}} \geq L_\alpha$ and $n_\alpha \geq L_{\tilde{\alpha}}$.

v) $C_\alpha \geq c_\alpha$.

Proof. i) In **Section 2.3**, we showed that there exists a bijection between quiver representations, their ranks and associated multisegments up to a change in the arbitrary labelling. Therefore if two multisegments α, β have isomorphic quiver representations, then there must also exist a bijection between the integers contained inside of the multisegments α and β. This bijection must preserve the number of minimal formation of $\cup_{\Delta\in\alpha}\Delta$ hence $c_\alpha = c_\beta$, and finally it must also preserve the lengths of these segments so $S_\alpha = S_\beta$.

ii) Taking the union intersection and conjunction of two segments can only increase the length of the resulting segment so $L_\alpha \leq L_\beta$.

iii) Taking the union intersection and conjunction of two segments either replaces the two segments with one or two segments, thus $n_\alpha \geq n_\beta$.

iv) If we construct the dual of α, then each element inside of the largest segment must be mapped into a different segment in the dual, therefore $n_{\tilde{\alpha}} \geq L_\alpha$ and by duality $n_\alpha \geq L_{\tilde{\alpha}}$.

v) c_α denotes the minimum number of segments in which $\cup_{\Delta\in\alpha}\Delta$ can be broken into. Thus we can decompose α into α_i's such that each α contains all Δ which correspond to a component formed by $\cup_{\Delta\in\alpha}\Delta$. Clearly we will thus have $\alpha = \bigsqcup_i \alpha_i$, and since $\tilde{\alpha}$ is formed by the Mœglin-Waldspurger algorithm and the precedes condition, then the dual of α will be formed by the dual of each α_i, so $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}$. Therefore $C_\alpha \geq c_\alpha$. □

Lemma 4.2.3. *If $L_{\tilde{\alpha}} = n_\alpha$, $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, then $n_\alpha = n_\beta = L_{\tilde{\alpha}} = L_{\tilde{\beta}}$.*

Proof. Using ii), iv) and iii) from **Lemma 4.2.2** followed by the assumption, we find that

$$L_{\tilde{\alpha}} \leq L_{\tilde{\beta}} \leq n_\beta \leq n_\alpha = L_{\tilde{\alpha}},$$

therefore

$$n_\alpha = n_\beta = L_{\tilde{\alpha}} = L_{\tilde{\beta}}.$$ □

Note by duality we also have

$$n_{\tilde{\alpha}} = n_{\tilde{\beta}} = L_\alpha = L_\beta.$$

Lemma 4.2.4. *Let α be an irreducible multisegment then*

$$n_{\tilde{\alpha}} \geq S_\alpha - n_\alpha + 1.$$

Proof. Each of the S_α-distinct values, x, from $\cup_{\Delta\in\alpha}\Delta$ must appear at least once as the maximum value of the segments in the dual, unless, x is one less than the minimum value from a segment. To see this, notice that x will be used as one of the maximum values, except when the final occurrence of $x+1$ is removed by the algorithm, and following this iteration x no longer appears in the multisegment. For this to occur $x+1$ must be the minimum value of a segment, otherwise, x would still appear in the segments whenever an $x+1$ is removed.

The number of minimum values of a segment is exactly n_α, however there are only $n_\alpha - 1$ possible maximum values which can be missed this way. Therefore the result follows, since we know at least $S_\alpha - (n_\alpha - 1)$ segments must start with distinct values, hence there must be at least $S_\alpha - n_\alpha + 1$ in $\tilde{\alpha}$. $\square$

Lemma 4.2.5. *Let* α *be an arbitrary multisegment then*

$$n_{\tilde{\alpha}} + n_\alpha \geq S_\alpha + C_\alpha.$$

Proof. Let

$$\alpha = \alpha_1 \sqcup \alpha_2 \sqcup \cdots \sqcup \alpha_{C_\alpha}$$

be a maximal irreducible decomposition of α. Each α_i will be an irreducible multisegment for $1 \leq i \leq C_\alpha$. so by **Lemma 4.2.4** we have

$$n_{\alpha_i} + n_{\widetilde{\alpha_i}} \geq S_{\alpha_i} + 1.$$

If we now sum over the i's then we find

$$\sum_{i=1}^{C_\alpha} n_{\alpha_i} + n_{\widetilde{\alpha_i}} = n_\alpha + n_{\tilde{\alpha}} \geq S_\alpha + C_\alpha = \sum_{i=1}^{C_\alpha} S_{\alpha_i} + 1.$$ $\square$

Further, if we fix some of the inherent properties of α then we can find more relations.

Definition 4.2.6. A multisegment α is *simple* if it has the form:

$$\alpha = \{[b, \ldots, e], [b+1, \ldots, e+1], \ldots, [b+n-1, \ldots, e+n-1]\}.$$

We saw an example of a simple multisegment in **Example 3.1.4**.

Proposition 4.2.7. *If* α *is a simple multisegment then we can verify:*

1. *The dual of* α *is also simple.*

2. $n_{\tilde{\alpha}} = L_\alpha$.

3. $c_\alpha = 1$.

4. $S_\alpha = n_\alpha + n_{\tilde{\alpha}} - c_\alpha = n_\alpha + L_\alpha - 1$.

Proof. Given a simple multisegment

$$\alpha = \{[b,\dots,e],[b+1,\dots,e+1],\dots,[b+n-1,\dots,e+n-1]\},$$

then its dual will be

$$\tilde{\alpha} = \{[b,\dots,b+n-1],[b+1,\dots,b+n],\dots,[e,\dots,e+n-1]\}.$$

1. Simply studying the form of $\tilde{\alpha}$ shows us that it is also simple.

2. The length of every segment in α is $L_\alpha = e-b+1$, and the number of segments in the dual is $n_{\tilde{\alpha}} = e-b-1$, thus $L_\alpha = n_{\tilde{\alpha}}$.

3. $\cup_{\Delta\in\alpha}\Delta = [b,e+n-1]$, therefore $c_\alpha = 1$.

4. $S_\alpha = (e+n-1)-b+1 = (e-b+1)+n-1 = L_\alpha + n_\alpha - 1 = n_{\tilde{\alpha}} + n_\alpha - c_\alpha$. □

Lemma 4.2.8. *If α is simple, $\alpha \le \beta$ and $L_\alpha = L_\beta$, then $\alpha = \beta$.*

Proof. Firstly, the length of every segment in α is given by L_α. If we perform either the union intersection or conjunction to form β then it will increase the length of a segment, which will result in increase the length of the longest segment. This would contradict the fact that $L_\alpha = L_\beta$. Consequently, the only way to ensure $L_\alpha = L_\beta$ is to not perform union intersection or conjunction, hence leave α unchanged so $\alpha = \beta$. □

Theorem 4.2.9. *Let α be a simple multisegment. If β is a multisegment such that $\alpha \le \beta$ and $\tilde{\alpha} \le \tilde{\beta}$, then $\alpha = \beta$.*

Proof. Firstly, α is simple so by **Proposition 4.2.7** ii) $L_\alpha = n_{\tilde{\alpha}}$, and hence by **Lemma 4.2.3** $L_\alpha = L_\beta$. Therefore **Lemma 4.2.8** states that $\alpha = \beta$. □

Therefore we have proved that the partial ordering relation will be satisfied for all simple multisegments.

Remark 4.2.10. The same will not be true for the case in which β is simple. In other words, if β is a simple multisegment then for any α such that $\alpha \le \beta$ and $\tilde{\alpha} \le \tilde{\beta}$ then it will not necessarily be true that $\alpha = \beta$. This will be illustrated in the example which follows.

Example 4.2.11. Let us consider the multisegments

$$\alpha = \{[0,1][1],[2],[2,3]\} \text{ and } \beta = \{[0,1],[1,2],[2,3]\}.$$

The associated dual multisegments to α and β are

$$\tilde{\alpha} = \{[0][1,2],[1,3]\} \text{ and } \tilde{\beta} = \{[0,2],[1,3]\}.$$

Therefore, we have the conditions that α and β are multisegments such that β is simple, $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, but $\alpha \neq \beta$.

4.2.1 Ladder Multisegments

We will now study broader family of multisegments for which there exists a natural ordering between each of the segments.

Definition 4.2.12 ([12])**.** We say that a multisegment α is a *ladder* if it has the form:

$$\alpha = \{\Delta_1, \ldots, \Delta_{n_\alpha}\},$$

where if we write $\Delta_i = [b_i, e_i]$ then for each $i < j$ we must have $b_i < b_j$ and $e_i < e_j$.

We saw an example of a ladder multisegment in **Example 3.1.5**. Also note that any simple multisegment will be a ladder multisegment, so we have seen an additional example of a ladder multisegment in **Example 3.1.4**.

Following on from **Section 3.1** in which we defined the Mœglin-Waldspurger algorithm and the segment generated by an iteration of the algorithm by Δ', then let us now specify that $\Delta'(\alpha)$ will be the segment generated by the first iteration of the Mœglin-Waldspurger algorithm. Also let us denote $\alpha - \Delta'(\alpha)$ to be the multisegment produced by the algorithm following the removal of the elements chosen for $\Delta'(\alpha)$, this will also be the multisegment in which the next iteration of the algorithm is carried out on. Thus the dual multisegment will be recursively generated by

$$\tilde{\alpha} = \{\Delta'(\alpha), \widetilde{\alpha - \Delta'(\alpha)}\}.$$

Proposition 4.2.13. *If α is a ladder multisegment then we can verify:*

1. $\alpha - \Delta'(\alpha)$ *is a ladder multisegment.*

2. $\tilde{\alpha}$ *is a ladder multisegment.*

3. *Let $x+1$ be an element chosen from Δ_{m+1} by the Mœglin-Waldspurger algorithm, then if x is in Δ_m it will be chosen to be in the same segment of the dual, otherwise, $x+1$ is the minimum value of the segment in the dual.*

4. *Let $x-1$ be an element chosen from Δ_m by the Mœglin-Waldspurger algorithm, then if x is in Δ_{m+1} it will be chosen to be in the same segment of the dual, otherwise, $x-1$ is the maximum value of the segment in the dual.*

5. *The irreducible decomposition of a ladder multisegment is unique and has exactly $C_\alpha = c_\alpha$ components.*

Proof. 1. The maximum value in α is e_{n_α}. The Mœglin-Waldspurger algorithm is such that it will choose the longest segment $(e_k, \ldots, e_{n_\alpha})$ such that $e_{i+1} = e_i + 1$ for $k \leq i \leq n_\alpha - 1$ and $e_i < e_{i+1}$ for $1 \leq i \leq k-1$. Thus we only need to check $\alpha - \Delta'(\alpha)$ satisfies the requirements for a ladder multisegment for $k-1 \leq i$. Notice that for any j and each $i < j$, $b_i < b_j$ by the initial requirements. This will trivially remain satisfied since either $b_i = e_i$ and e_i is chosen by the algorithm at which point the segment is completely removed so won't need to conform to the requirements; otherwise b_i will remain unchanged and continue to satisfy the inequality. For $1 \leq i \leq k-1$ the e_i's will remain unchanged so continue to satisfy $e_i < e_{i+1}$. Further, since $e_i < e_i + 1 = e_{i+1}$ for $k \leq i \leq n_\alpha - 1$, and both e_i, e_{i+1} are reduced by one then $e_i - 1 < e_{i+1} - 1$ will remain. Finally, e_{k-1} will remain the same but e_k will be reduced by one, however by the construction and the Mœglin-Waldspurger algorithm $e_{k-1} < e_k$ and $e_{k-1} + 1 \neq e_k$ so $e_{k-1} + 1 < e_k, e_{k-1} < e_k - 1$ which shows the requirement remains satisfied following the iteration. Therefore for each $i < j$, $b_i < b_j$ and $e_i < e_j$ for $\alpha - \Delta'(\alpha)$, so $\alpha - \Delta'(\alpha)$ is ladder multisegment.

2. At each iteration of the Mœglin-Waldspurger algorithm on the ladder multisegment the maximum value of α will be chosen to end the new segment. By construction of a ladder multisegment, the multiplicity of this maximum value in the multisegment is one, therefore during any

subsequent iteration following its removal the end value must be less than it. Also 1. states that following each iteration a ladder multisegment will be retrieved, so the end value of an iteration is always greater than the end value of the following iteration.

Similarly, if $[e_k, e_{m_\alpha}]$ was constructed at one iteration of the algorithm then at the following iteration we have two cases:

(a) The case in which $\Delta_{k+1}, \ldots, \Delta_{m_\alpha}$ were all originally singletons, hence the end value of the segment generated will be less than the base value e_k chosen during the previous iteration.

(b) Otherwise, let Δ_i denote the highest segment which was not originally a singleton in α then $k+1 \leq i \leq m_\alpha$, then $e_i - 1$ will be chosen to end the new segment. By **Proposition 3.1.9**, the segments chosen at each iterative stage must be of increasing length, hence all of the segments $\Delta_k, \ldots \Delta_i$ will still be present with end values one less than at the previous stage. Thus a segment can be constructed containing $e_k - 1, \ldots, e_i - 1$, so the base value of the new segment will be less than $e_k - 1$ and in turn be less than the base value of the segment generated in the previous iteration.

Therefore at each iteration both the base and end values will be less than those chosen in the previous iteration, so a ladder multisegment will be constructed.

3. Let $x+1$ be an element chosen from Δ_{m+1} by the Mœglin-Waldspurger algorithm. Note that Δ_{m+1} only precedes the segments Δ_k for $k \leq m$, since $b_k < b_{m+1}$ only if $k \leq m$. By (1), $\alpha - \Delta'(\alpha)$ is also a ladder multisegment and the Mœglin-Waldspurger algorithm preserves the ordering of the ladder multisegment. Since $x+1$ is chosen from Δ_{m+1} then for this iteration $e_{m+1} = x+1$ and $e_m < e_{m+1} = x+1$. Therefore $e_m \leq x$ so let us consider the two cases:

   (a) If $e_m = x$ then it must be chosen to be in the same segment, since $e_k < e_m = x$ for all $k < m$.

   (b) If $e_m < x$ then $x+1$ is the minimum value of the segment in the dual, since $e_k < e_m < x$ for all $k < m$ and Δ_k only precedes Δ_{m+1} for $k \leq m$.

4. Let $x-1$ be an element chosen from Δ_m by the Mœglin-Waldspurger algorithm. Note that Δ_m can only be preceded by segments Δ_k for $k \geq m+1$, since $b_m < b_k$ only if $k \geq m+1$. By

(1), $\alpha - \Delta'(\alpha)$ is also a ladder multisegment and the Mœglin-Waldspurger algorithm preserves the ordering of the ladder multisegment. Since $x-1$ is chosen from Δ_m then for this iteration $e_m = x-1$ and $x-1 = e_m < e_{m+1} < e_j$ for $j > m+1$. Therefore when $e_m = x-1$ then the only possibility in which $x-1$ is not the maximum value of a segment is when $e_{m+1} = x$, hence x is contained in Δ_{m+1}. Since the Mœglin-Waldspurger chooses the maximum values of segments in descending order then $x-1$ can only be chosen to start a segment if the algorithm has already chosen every x. Therefore either $x \in \Delta_{m+1}$ so for some iteration $e_{m+1} = x$ and hence $x-1 \in \Delta_m$ will be chosen to be in the same segment, otherwise, $x \notin \Delta_{m+1}$ so $x-1 \in \Delta_m$ will be chosen to be the maximum value of a segment.

5. Firstly c_α is the minimum number of segments which form $\cup_{\Delta\in\alpha}\Delta$, thus for each connected segment let us denote it by Δ_i. Then let us define α_i to be the sub-multisegments that consist of all the segments Δ in α such that the union of these Δ's form Δ_i. By construction there will be c_α number of α_i's and every segment must be contained in exactly one α_i, so $\alpha = \bigsqcup_i \alpha_i$.

   Given two distinct components α_j and α_k, where $\cup_{\Delta\in\alpha_j}\Delta = [b_j, e_j]$ and $\cup_{\Delta\in\alpha_k}\Delta = [b_k, e_k]$. Then without loss of generality let α_j and α_k be such that $e_j < b_k$. Also note that $[b_j, e_j] \cap [b_k, e_k] = \emptyset$ so $e_j < b_k + 1$, otherwise, $\cup_{\Delta\in\alpha_j\sqcup\alpha_k}\Delta$ would form a single segment and c_α would not be the smallest value. Now since $e_j < b_k + 1$, then for any segments Δ_j in α_j and Δ_k in α_k, Δ_k will not precede Δ_j, so there will be no interaction between different components in the Mœglin-Waldspurger algorithm. Thus $\tilde{\alpha} = \bigsqcup_i \widetilde{\alpha_i}$, since the Mœglin-Waldspurger algorithm will only have to consider preceding elements inside of the individual components.

   Let us assume that $c_\alpha < C_\alpha$. Note that we have already shown that there is no interaction between blocks, so the only possibility is that we can break one of the components α_i into more components α_{i_1} and α_{i_2} and $\widetilde{\alpha_i} = \widetilde{\alpha_{i_1}} \sqcup \widetilde{\alpha_{i_2}}$. Note that α_i is a ladder multisegment, so $\alpha_i = \left\{\Delta_1, \ldots, \Delta_{n_{\alpha_i}}\right\}$. By construction for some $m = [1, \ldots, n_{\alpha_i} - 1]$, there will be $\Delta_m = [b_m, e_m] \in \alpha_{i_1}$ and $\Delta_{m+1} = [b_{m+1}, e_{m+1}] \in \alpha_{i_2}$. Recall that $b_m < b_{m+1}$, $e_m < e_{m+1}$ and since $\cup_{\Delta\in\alpha_i}\Delta$ forms a single segment then $b_{m+1} \leq e_m + 1$, therefore Δ_m will precede Δ_{m+1}. Also note that $e_m + 1$ is an element in Δ_{m+1} because $e_m < e_{m+1}$, therefore by (3) when the Mœglin-Waldspurger algorithm selects $e_m + 1$ from Δ_{m+1} then e_m will be selected from Δ_m to be in the same segment of the dual. Instead, let us consider $\widetilde{\alpha_{i_1}}$ and $\widetilde{\alpha_{i_2}}$. Note Δ_{m+1} will only precede

Δ_k for $k \leq m$. However, $e_k < e_m$ for $k < m$ thus $e_m + 1$ will be the minimum value of the segment in the dual of α_{i_2} by (3). Similarly, note that $b_{m+1} - 1$ is an element in Δ_m because $b_m < b_{m+1}$, therefore by (4) when the Mœglin-Waldspurger algorithm selects $b_{m+1} - 1$ from Δ_m then b_{m+1} will be selected from Δ_{m+1} to be in the same segment of the dual. Instead, let us consider $\widetilde{\alpha_{i_1}}$ and $\widetilde{\alpha_{i_2}}$. Note Δ_m is only preceded by Δ_k for $k \geq m+1$. However, $b_k > b_{m+1}$ for $k > m+1$ thus $b_{m+1} - 1$ will be the maximum value of the segment in the dual of α_{i_1} by (4). Therefore $\widetilde{\alpha}_i \neq \widetilde{\alpha_{i_1}} \sqcup \widetilde{\alpha_{i_2}}$, hence we have a contradiction so $C_\alpha = c_\alpha$. □

Lemma 4.2.14. *If $n_{\tilde{\alpha}} + n_\alpha = S_\alpha + C_\alpha$, $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$ then $n_{\tilde{\alpha}} = n_{\tilde{\beta}}$, $n_\alpha = n_\beta$ and $n_{\tilde{\beta}} + n_\beta = S_\beta + C_\beta$.*

Proof. Using **Lemma 4.2.2** iii) we know that $n_\alpha \geq n_\beta$ and $n_{\tilde{\beta}} \geq S_\beta - n_\beta + C_\beta$ by **Lemma 4.2.5**. Now using the assumed conditions, we find

$$n_{\tilde{\alpha}} \geq n_{\tilde{\beta}} \geq S_\beta - n_\beta + C_\beta \geq S_\alpha - n_\alpha + C_\alpha = n_{\tilde{\alpha}}.$$

Therefore we find $n_{\tilde{\alpha}} = n_{\tilde{\beta}}$ and $n_{\tilde{\beta}} = S_\beta - n_\beta + C_\beta$, hence $n_{\tilde{\beta}} + n_\beta = S_\beta + C_\beta$. Finally, $S_\alpha - n_\alpha + C_\alpha = S_\beta - n_\beta + C_\beta$ which implies $n_\alpha = n_\beta$ since $S_\alpha = S_\beta$ and $C_\alpha = C_\beta$ by **Lemma 4.2.2** i) and **Proposition 4.2.13** v). □

Lemma 4.2.15. *If α and β are ladder multisegments such that $\alpha \leq \beta$ and $n_\alpha = n_\beta$, then $\alpha = \beta$.*

Proof. Recall that in order to obtain β we perform elementary operations on two segments Δ_1 and Δ_2. When we perform these operations then we choose one of the three: union intersection, conjunction or leave α unchanged. Notice that conjunction will only replace the two segments with one whereas the other two will keep two segments. Thus we cannot perform conjunction since we have assumed $n_\alpha = n_\beta$.

If we instead perform union intersection and let $\Delta_3 = \Delta_1 \cap \Delta_2$ and $\Delta_4 = \Delta_1 \cup \Delta_2$. However $\Delta_3 \subset \Delta_4$, therefore a single operation of union intersection will not form a β such that it is a ladder multisegment. It could however be the case that multiple union intersections can be performed to form such a β. To do this let us take another union intersection but this time it must involve Δ_3, since we need to break the condition $\Delta_3 \subset \Delta_4$. So let us perform union intersection on Δ_3 and another segment Δ_5 contained in α, then $\Delta_3' = \Delta_3 \cap \Delta_5$ and $\Delta_4' = \Delta_3 \cup \Delta_5$. However, once again we find that $\Delta_3' \subset \Delta_3 \subset \Delta_4$, which again forms a multisegment which is not a ladder multisegment. Therefore by recursion, no matter how many iterations of union intersection are performed it will not be possible to recover the ladder multisegment property for β. So neither union intersection or conjunction can be performed on α, hence $\alpha = \beta$. □

Lemma 4.2.16. *Suppose α is a multisegment and that $x \in \Delta'(\alpha)$ and that the occurrence of x in $\Delta'(\alpha)$ is selected from a singleton of α, then for all $y \in \Delta'(\alpha)$ such that $y \geq x$ we have that y is also selected from a singleton of α.*

Proof. This follows from **Proposition 3.1.9**, since preceding segments must be chosen in increasing length and if x is chosen from a singleton Δ_x then for each $y \geq x$ the segments Δ_y chosen prior must also be singletons. □

Lemma 4.2.17. *Suppose α is a multisegment such that*

$$x \in \left(\bigcup_{\Delta\in\alpha} \Delta\right) \Big\backslash \left(\bigcup_{\Delta\in\alpha-\Delta'(\alpha)} \Delta\right),$$

then for $y \geq x+1$ we have that y is selected from a singleton.

Proof. Let Δ be the segment in α from which we select $x+1$, then $x \notin \Delta$ or otherwise

$$x \notin \left(\bigcup_{\Delta\in\alpha} \Delta\right) \Big\backslash \left(\bigcup_{\Delta\in\alpha-\Delta'(\alpha)} \Delta\right).$$

So it follows that Δ is a singleton and by **Lemma 4.2.16** each $y \geq x+1$ is a singleton. □

Lemma 4.2.18. *Let α be an irreducible multisegment then*

$$S_\alpha - S_{\alpha-\Delta'(\alpha)} \leq \min\left\{n_\alpha - n_{\alpha-\Delta'(\alpha)} + c_{\alpha-\Delta'(\alpha)}, n_\alpha - n_{\alpha-\Delta'(\alpha)} + 1\right\},$$

with equality if none of the deleted segments are contained in other segments.

Proof. The value $S_\alpha - S_{\alpha-\Delta'(\alpha)}$ counts the number of distinct elements x where

$$x \in \left(\bigcup_{\Delta\in\alpha} \Delta\right) \Big\backslash \left(\bigcup_{\Delta\in\alpha-\Delta'(\alpha)} \Delta\right),$$

that is, it counts the number of elements which are completely removed following a single iteration.

Lemma 4.2.17 states that all but the smallest of these comes from deleting singletons from α, this contributes $n_\alpha - n_{\alpha-\Delta'(\alpha)} + 1$ to the formula. However if $c_{\alpha-\Delta'(\alpha)} = 0$ then in fact they all came from singletons. Therefore strict inequality holds if any deleted segments are contained in another segment and equality if none are. $\square$

Lemma 4.2.19. *If α is an irreducible ladder multisegment then*

$$n_{\tilde{\alpha}} + n_\alpha = S_\alpha + c_\alpha.$$

Proof. We will proceed by induction on S_α, the result is trivial when $S_\alpha = 0$, so we assume that $S_\alpha > 0$ and hence $C_\alpha = c_\alpha = 1$. The first step of the Mœglin-Waldspurger algorithm is to construct a segment $\Delta'(\alpha)$ that ends at e_α. We then apply the algorithm recursively to the multisegment $\alpha - \Delta'(\alpha)$. It is important to note that the multisegment $\alpha - \Delta'(\alpha)$ will also be a ladder multisegment and following **Lemma 4.2.18**

$$S_\alpha - S_{\alpha-\Delta'(\alpha)} = \min\left\{n_\alpha - n_{\alpha-\Delta'(\alpha)} + c_{\alpha-\Delta'(\alpha)}, n_\alpha - n_{\alpha-\Delta'(\alpha)} + 1\right\},$$

since it is irreducible and further it is a ladder multisegment so $c_{\alpha-\Delta'(\alpha)} \leq 1$. Therefore,

$$S_\alpha - S_{\alpha-\Delta'(\alpha)} = n_\alpha - n_{\alpha-\Delta'(\alpha)} + c_{\alpha-\Delta'(\alpha)},$$

and by the inductive hypothesis

$$\begin{aligned} n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_{\alpha-\Delta'(\alpha)} =& S_{\alpha-\Delta'(\alpha)} + c_{\alpha-\Delta'(\alpha)}, \\ =& (S_\alpha - n_\alpha + n_{\alpha-\Delta'(\alpha)} - c_{\alpha-\Delta'(\alpha)}) + c_{\alpha-\Delta'(\alpha)}, \\ =& S_\alpha - (n_\alpha - n_{\alpha-\Delta'(\alpha)}). \end{aligned}$$

Therefore $S_\alpha = n_\alpha + n_{\widetilde{\alpha-\Delta'(\alpha)}}$. Given that $n_{\widetilde{\alpha-\Delta'(\alpha)}} = n_{\tilde{\alpha}} - 1 = n_{\tilde{\alpha}} - c_\alpha$, we combine the two equations to find that $n_{\tilde{\alpha}} + n_\alpha = S_\alpha + c_\alpha$ as required. $\square$

Corollary 4.2.20. *For an arbitrary ladder multisegment α, we have $n_{\tilde{\alpha}} + n_\alpha = S_\alpha + c_\alpha$.*

Lemma 4.2.21. *If* α *is any multisegment and*

$$n_{\tilde{\alpha}} + n_{\alpha} = S_{\alpha} + c_{\alpha},$$

then α *is a ladder multisegment.*

Proof. We shall prove the contrapositive of the statement, that is, if α is not a ladder multisegment then

$$n_{\tilde{\alpha}} + n_{\alpha} > S_{\alpha} + c_{\alpha}.$$

We shall again proceed by induction on the total numbers of points x in α. If $C_{\alpha} > 1$ then we decompose it into irreducible components and consider

$$n_{\tilde{\alpha}} + n_{\alpha} = \sum_i n_{\tilde{\alpha_i}} + n_{\alpha_i} \geq \sum_i (S_{\alpha_i} + C_{\alpha_i}) \geq S_{\alpha} + C_{\alpha}.$$

Then either:

1. At least one irreducible component is not a ladder multisegment, in which case we obtain the result immediately by induction as the first inequality will be strict.

2. At least two of the irreducible components overlap in which case

   $$\sum_{i=1}^{C_{\alpha}} S_{\alpha_i} > S_{\alpha},$$

   so the second inequality will be strict.

Therefore we are thus reduced to the case $C_{\alpha} = 1$ which implies $c_{\alpha} = 1$.

First, recall that **Lemma 4.2.18** states for an irreducible multisegment

$$S_{\alpha} - S_{\alpha - \Delta'(\alpha)} \leq \min\left\{ n_{\alpha} - n_{\alpha - \Delta'(\alpha)} + c_{\alpha - \Delta'(\alpha)}, n_{\alpha} - n_{\alpha - \Delta'(\alpha)} + 1 \right\},$$

however since α is not a ladder multisegment, $C_{\alpha - \Delta'(\alpha)} > 0$ then

$$S_{\alpha} - S_{\alpha - \Delta'(\alpha)} \leq n_{\alpha} - n_{\alpha - \Delta'(\alpha)} + 1.$$

Suppose that $\alpha - \Delta'(\alpha)$ is not a ladder multisegment, then we apply the inductive hypothesis to $\alpha - \Delta'(\alpha)$ which is smaller than α,

$$n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_{\alpha-\Delta'(\alpha)} > S_{\alpha-\Delta'(\alpha)} + C_{\alpha-\Delta'(\alpha)} \geq S_\alpha - 1 - (n_\alpha - n_{\alpha-\Delta'(\alpha)}) + C_{\alpha-\Delta'(\alpha)},$$

so

$$n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_\alpha + 1 > S_\alpha + C_{\alpha-\Delta'(\alpha)}.$$

Further $n_{\tilde{\alpha}} = n_{\widetilde{\alpha-\Delta'(\alpha)}} + 1$, so we find $n_{\tilde{\alpha}} + n_\alpha > S_\alpha + C_{\alpha-\Delta'(\alpha)}$, as required.

Instead let us consider $\alpha - \Delta'(\alpha)$ is a ladder multisegment, and let us show that the following inequality is strict

$$S_\alpha - S_{\alpha-\Delta'(\alpha)} \leq n_\alpha - n_{\alpha-\Delta'(\alpha)} + 1.$$

Since α is not a ladder multisegment then we have $\Delta_1 = [b_1, e_1]$ and $\Delta_2 = [b_2, e_2]$ in which at least one of the following cases occurs:

1. $b_1 > b_2$ and $e_1 < e_2$:
   (a) $b_1 = e_1$,
   (b) $b_1 \neq e_1$.
2. $e_1 = e_2$:
   (a) $b_1 = e_1$,
   (b) $b_2 = e_2$,
   (c) $b_1 \neq e_1$ and $b_2 \neq e_2$.
3. $b_1 = b_2$ and $e_1 \neq e_2$:
   (a) $b_1 = e_1$,
   (b) $b_2 = e_2$,
   (c) $b_1 \neq e_1$ and $b_2 \neq e_2$.

For $\alpha - \Delta'(\alpha)$ to be a ladder multisegment, it must have selected either e_1 or e_2 from one of the segments. Thus we can consider each case separately:

1. If e_2 becomes smaller and e_1 does not then the result will not be a ladder multisegment, so let us assume e_1 is selected
   (a) If we select Δ_1 in the algorithm we delete the segment, however the segment Δ_2 contains e_1, so $S_{\alpha-\Delta'(\alpha)}$ does not decrease and hence by **Lemma 4.2.18** the inequality will be strict.
   (b) Even though $e_1' < e_1$ we will remain in case 1) thus the result will not be a ladder multisegment.

2. In this case the algorithm either selects Δ_1 or Δ_2, but not both:

   (a) If we select Δ_1, then we delete Δ_1. Since $e_1 \in \Delta_2 = \Delta_2'$ so by **Lemma 4.2.18** $S_{\alpha-\Delta'(\alpha)}$ will not decrease so the inequality will be strict. Instead, if we select Δ_2 then it is because Δ_2 is preceded by a Δ_3 which does not precede Δ_1. However $\Delta_1 \subseteq \Delta_3'$ hence the resulting multisegment will not be a ladder multisegment.

   (b) Identical to (a) by symmetry.

   (c) If $b_1 = b_2$ then regardless of whether Δ_1 or Δ_2 is chosen the resulting $\alpha - \Delta'(\alpha)$ will not be a ladder multisegment. Without loss of generality, suppose $b_1 \leq b_2$. If Δ_2 is selected then $\Delta_2' \subseteq \Delta_1' = \Delta_1$ which is not a ladder multisegment. Otherwise, Δ_1 will be selected, however that is only the case if Δ_3 is selected and precedes Δ_1 but not Δ_2. However, $\Delta_2 = \Delta_2' \subseteq \Delta_3'$, hence $\alpha - \Delta'(\alpha)$ will not be a ladder multisegment.

3. $b_1 = b_2$ and $e_1 \neq e_2$:

   (a) By symmetry we may assume $e_2 > e_1$. If we select Δ_1 then we delete Δ_1, however since $e_1 \in \Delta_2$ this will not decrease S_α so again by **Lemma 4.2.18** the inequality will be strict. Instead if we select Δ_2 and not Δ_1, then we will still have $\Delta_1' = \Delta_1 \subseteq \Delta_2'$. Hence it will not be a ladder multisegment.

   (b) Identical to (a) by symmetry.

   (c) As $b_1 < e_1$ and $b_2 < e_2$, then by any modification to Δ_1 and Δ_2 we will still have $b_1 = b_2 = b_1' = b_2'$. Therefore it will not be a ladder multisegment.

Therefore

$$S_\alpha - S_{\alpha-\Delta'(\alpha)} < n_\alpha - n_{\alpha-\Delta'(\alpha)} + 1,$$

and since $\alpha - \Delta'(\alpha)$ is ladder multisegment then by **Corollary 4.2.20** we have

$$n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_{\alpha-\Delta'(\alpha)} = S_{\alpha-\Delta'(\alpha)} + C_{\alpha-\Delta'(\alpha)},$$

and hence

$$\begin{aligned}
n_{\tilde{\alpha}} + n_{\alpha} &= n_{\alpha} + n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_{\alpha-\Delta'(\alpha)} - n_{\alpha-\Delta'(\alpha)} + 1, \\
&> S_{\alpha} - S_{\alpha-\Delta'(\alpha)} + n_{\widetilde{\alpha-\Delta'(\alpha)}} + n_{\alpha-\Delta'(\alpha)}, \\
&= S_{\alpha} - S_{\alpha-\Delta'(\alpha)} + S_{\alpha-\Delta'(\alpha)} + C_{\alpha-\Delta'(\alpha)}, \\
&= S_{\alpha} + C_{\alpha-\Delta'(\alpha)}.
\end{aligned}$$

However, α is not a ladder multisegment so $C_{\alpha-\Delta'(\alpha)} \geq 1$ which implies

$$n_{\tilde{\alpha}} + n_{\alpha} > S_{\alpha} + c_{\alpha}. \qquad \square$$

Corollary 4.2.22. *For an arbitrary multisegment α which is not a ladder multisegment, we have*

$$n_{\tilde{\alpha}} + n_{\alpha} > S_{\alpha} + C_{\alpha} \geq S_{\alpha} + c_{\alpha},$$

and hence a multisegment α is a ladder multisegment if and only if

$$n_{\tilde{\alpha}} + n_{\alpha} = S_{\alpha} + C_{\alpha} = S_{\alpha} + c_{\alpha}.$$

Proof. For a multisegment which is not a ladder multisegment either:

1. At least one irreducible component is not a ladder multisegment, in which case following the proof of **Lemma 4.2.21** we obtain the inequality.

2. Two of the ladder multisegments overlap in which case

$$\sum_{i=1}^{C_{\alpha}} (S_{\alpha_i} + 1) > S_{\alpha} + c_{\alpha}. \qquad \square$$

Theorem 4.2.23. *Let α be a ladder multisegment. If β is a multisegment such that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, then $\alpha = \beta$.*

Proof. Since α is a ladder multisegment then $n_{\tilde{\alpha}} + n_{\alpha} = S_{\alpha} + C_{\alpha}$ by **Corollary 4.2.22**. Using **Lemma 4.2.14**, $n_{\tilde{\beta}} + n_{\beta} = S_{\beta} + C_{\beta}$ and $n_{\alpha} = n_{\beta}$, thus β is a ladder multisegment by **Corollary 4.2.22**. Therefore we have satisfied all hypothesis of **Lemma 4.2.15** so $\alpha = \beta$. $\square$

Remark 4.2.24. At each iterative step of the Mœglin-Waldspurger algorithm the set of preceding segments which are chosen form an irreducible ladder multisegment. This follows from **Proposition 3.1.9** and the properties discussed throughout this subsection must therefore be satisfied for each iteration.

4.2.2 Arthur Type

We will now present the proof that another family of multisegments satisfy the partial ordering relation on multisegments and relate this to a significant conjecture in the local Langlands correspondence that

ABV-packets for orbits of Arthur type in GL_n are singletons.

To do this we must first introduce this notion of Arthur type. A *Langlands parameter of Arthur type* is a Langlands parameter *phi* such that $\phi = \phi_\psi$ (See **Definition 2.1.6**) as defined in the book [5, Section 3.6]. One property that this enforces is that the corresponding multisegments must have the property of being symmetric along the zero element i, where λ_i corresponds to the q^0-eigenspace. That is, if we relabel each element in the multisegment α to be such that $i \to 0$, $i-1 \to -1$, $i+1 \to 1$, $\ldots$; then the segment $\Delta = [b,e] \in \alpha$ if and only if the segment $-\Delta = [-e,-b] \in \alpha$. A key consequence of this restriction is that we are now only considering symmetric irreducible multisegments, so when S_α is odd then the description is trivial since 0 will be a central value. Alternatively when S_α is even, we have to slightly modify the description to be such that the two labelings each side of the symmetry will be $-\frac{1}{2}$ and $\frac{1}{2}$, and any subsequent values will then differ by 1 as they get further away from the centre. Note this description preserves the structure of their being 1 between each of the labelings of the eigenvectors, and hence preserves the previously discussed properties. Further, we defined the maximum value of our multisegment to be e_α so it will always be true that $S_\alpha = 2e_\alpha + 1$.

Example 4.2.25. Given the multisegment

$$\alpha = \{[-2,-1],[-1,1],[0,0],[1,2]\},$$

then α satisfies the conditions to be symmetric.

Alternatively, let us consider the multisegment

$$\beta = \{[-3,-2],[-2,2],[0,1],[2,3]\},$$

then β is not symmetric, since $\Delta = [0,1] \in \beta$ but $-\Delta = [-1,0] \notin \beta$.

The previous propositions and lemmas relating to the numerical invariants will remain satisfied for these specific families of symmetric multisegments. We will now extend our study of numerical invariants of multisegments to the family of symmetric multisegments.

One may expect that given a Langlands parameter with a corresponding symmetric multisegment α then the partial ordering relation will always be satisfied, however in the next example we will see that this will not always be the case. In fact, even with an additional restriction that the multisegment β must also be symmetric, then once again the following example will prove as a counter example for the partial ordering relation.

Example 4.2.26. Let us consider the multisegments

$$\alpha = \{[-1][-1,0],[0,1],[1]\} \;\text{ and }\; \beta = \{[-1],[0],[1],[-1,1]\}.$$

Then by studying their rank triangles (below) we can see that $\alpha \leq \beta$.

$$\begin{array}{ccccc} 2 & & 2 & & 2 \\ & 1 & & 1 & \\ & & 0 & & \end{array} \qquad\qquad \begin{array}{ccccc} 2 & & 2 & & 2 \\ & 1 & & 1 & \\ & & 1 & & \end{array}$$

Both α and β are self dual hence

$$\tilde{\alpha} = \{[-1][-1,0],[0,1],[1]\} \;\text{ and }\; \tilde{\beta} = \{[-1],[0],[1],[-1,1]\}.$$

Therefore we have the conditions that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$. Thus α and β are both symmetric and do not satisfy the partial ordering relation.

The restriction to only studying those Langlands parameters of Arthur type imposes a further condition on the multisegment α that α must be formed from the union of simple symmetric multisegments as discussed in [4, Remark 1.1].

Proposition 4.2.27. *Let α be any multisegment which is formed by taking the union of m simple symmetric multisegments $\alpha_1, \dots \alpha_m$. Then*

$$\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}.$$

Proof. Let us assume that $\alpha_1, \dots \alpha_m$ are simple symmetric multisegments, and $\alpha = \alpha_1 \sqcup \cdots \sqcup \alpha_m$. Then for all $1 \leq i \leq m$ the multisegments have the following forms

$$\alpha_i = \{[-e_i, -b_i], [-e_i+1, -b_i+1], \dots, [b_i-1, e_i-1], [b_i, e_i]\}.$$

Without loss of generality, we can assume α_1 contains the shortest segment containing the maximum value e and this segment is chosen first by Mœglin-Waldspurger algorithm, that is, e_1 from $[b_1, e_1]$ in α_1 is chosen first. Thus e_1 is the maximum value in the multisegment α. By the natural ordering of simple multisegments, the shortest possible preceding segments of $[b_1, e_1]$ ending in n will also be contained in α_1 for $-b_1 \leq n \leq e_1$, and hence will be chosen by the algorithm from α. Note $-b_1$ will be chosen from the segment $[-e_1, -b_1]$, and since e_1 is the maximum value of the multisegment then by symmetry $-e_1$ is the minimum value, thus there cannot exist a segment which precedes $[-e_1, b_1]$.

If each of the chosen segments were singletons, then α_1 is completely removed during that iteration, and hence a simple symmetric remains. Otherwise, **Theorem 3.1.8** states that both the Mœglin-Waldspurger and Alternate Mœglin-Waldspurger algorithms will compute the same duals. So let us now use the Alternate Mœglin-Waldspurger algorithm to compute the next iteration on $\alpha - \Delta'(\alpha)$. Firstly, $-e_1$ will be the minimum value of the multisegment α due to symmetry and since it is not removed from the segment $[-e_1, -b_1]$ during the first iteration, then it will still be the minimum value of $\alpha - \Delta'(\alpha)$. Also $[-e_1, -b_1 - 1]$ will be the shortest segment in $\alpha - \Delta'(\alpha)$ containing $-e_1$, since by symmetry $[-e_1, -b_1]$ was the shortest segment containing $-e_1$ in α, and following the first iteration each segment either remains the same length or gets shorter by one. By the natural ordering of simple multisegments, the shortest possible proceeding segments of $[-e_1, -b_1]$ with base value n will also be contained in α_1 for $-e_1 \leq n \leq b_1$, and hence will be chosen by the algorithm from

α. Note b_1 will be chosen from the segment $[b_1, e_1 - 1]$, and since the maximum value of the multisegment $\alpha - \Delta'(\alpha)$ is less than or equal to e_1. Then the only possible case for a preceding segment is if there exists $2 \le i \le m$ such that $e_i = e_1$, however by the first iteration of the algorithm $b_i \le b_1$, since $[b_1, e_1]$ must be either the same length or shorter than $[b_i, e_i]$ when $e_i = e_1$. Hence there will be no preceding segment since $b_i \le b_1$.

Therefore the first two iterations of these Mœglin-Waldspurger algorithms construct the segments $[-b_1, e_1]$ and $[-e_1, b_1]$. Notice that these iterations work exclusively on the segments of α_1, and hence construct the two segments are both contained in $\widetilde{\alpha_1}$, whilst leaving $\alpha_2, \dots, \alpha_m$ unchanged. Also the construction of these two segments will remove the highest and lowest values from each segment of α_1 during the first and second iterations respectively. Note if during the first iteration all singletons are chosen then we can just ignore the second iteration. Let us denote α'' to be the multisegment remaining from α following these two iterations, then

$$\alpha'' = \{[-e_1+1, -b_1-1], [-e_1+2, -b_1], \dots, [b_1, e_1-2], [b_1+1, e_1-1]\} \sqcup \alpha_2 \sqcup \dots \sqcup \alpha_m,$$

so let us denote the multisegment

$$\alpha_1'' = \{[-e_1+1, -b_1-1], [-e_1+2, -b_1], \dots, [b_1, e_1-2], [b_1+1, e_1-1]\}.$$

Thus α_1'' will be a simple multisegment, and naturally it also satisfies the symmetric property.

Thus the remaining multisegment $\alpha'' = \alpha_1'' \sqcup \alpha_2 \sqcup \dots \sqcup \alpha_m$, where $\alpha_1'', \alpha_2, \dots, \alpha_m$ are all simple symmetric multisegments. So we can invoke a recursive argument, and conclude that at each pair of iterations will form dual segments exclusively a single simple symmetric multisegment, and that the remaining multisegment following the iterations will be a union of simple symmetric multisegments. Further, the duals of the original multisegments $\alpha_1, \dots, \alpha_m$ will thus be computed independently, since for all $1 \le i \le m$

$$\widetilde{\alpha_i} = \left\{\Delta'(\alpha_i), \widetilde{\alpha_i - \Delta'(\alpha_i)}\right\} = \left\{\Delta'(\alpha_i), \Delta'(\alpha_i - \Delta'(\alpha_i)), \widetilde{\alpha_i''}\right\} = \left\{[-b_i, e_i], [-e_i, b_i], \widetilde{\alpha_i''}\right\}.$$

Therefore $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \dots \sqcup \widetilde{\alpha_m}$. □

Corollary 4.2.28. *Let α be any symmetric multisegment which is formed by taking the union of two simple symmetric multisegments α_1 and α_2. Then*

$$\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{\alpha_2}.$$

To begin our study of the partial ordering relation for Langlands parameters of Arthur type, we will first study the case in which α is formed by the union of two simple symmetric multisegments.

Proposition 4.2.29. *If α is a multisegment formed by the union of two simple symmetric multisegments α_1 and α_2, then we can verify:*

1. *The dual of α will also be formed by the union of two simple symmetric multisegments.*

2. $n_{\tilde{\alpha}} = L_{\alpha_1} + L_{\alpha_2}$.

3. $L_{\tilde{\alpha}} = \max\{n_{\alpha_1}, n_{\alpha_2}\}$.

4. $c_{\alpha} = 1$.

5. $S_{\alpha} = \max\{S_{\alpha_1}, S_{\alpha_2}\} = \max\{n_{\alpha_1} + L_{\alpha_1} - 1, n_{\alpha_2} + L_{\alpha_2} - 1\}$.

Proof.

1. **Corollary 4.2.28** states that $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{\alpha_2}$, and by **Proposition 4.2.7** both $\widetilde{\alpha_1}$ and $\widetilde{\alpha_2}$ will also be simple. Following the proof of **Corollary 4.2.28**, both $\widetilde{\alpha_1}$ and $\widetilde{\alpha_2}$ will be symmetric. Thus the dual of α will be formed by the union of two simple symmetric multisegments $\widetilde{\alpha_1}$ and $\widetilde{\alpha_2}$.

2. Firstly, $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{\alpha_2}$ by **Corollary 4.2.28**, thus $n_{\tilde{\alpha}} = n_{\widetilde{\alpha_1}} + n_{\widetilde{\alpha_2}}$. Note α_1 and α_2 are both simple thus by **Proposition 4.2.7** $n_{\widetilde{\alpha_1}} = L_{\alpha_1}$ and $n_{\widetilde{\alpha_2}} = L_{\alpha_2}$, so $n_{\tilde{\alpha}} = L_{\alpha_1} + L_{\alpha_2}$.

3. $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{\alpha_2}$ by **Corollary 4.2.28**, therefore the segment of maximum length either results from $\widetilde{\alpha_1}$ or $\widetilde{\alpha_2}$. Hence

$$L_{\tilde{\alpha}} = \max\left\{L_{\widetilde{\alpha_1}}, L_{\widetilde{\alpha_2}}\right\} = \max\{n_{\alpha_1}, n_{\alpha_2}\},$$

by **Proposition 4.2.7**.

4. Let us assume that e is the maximum value of α then by symmetry $-e$ will be the minimum value, thus there are three possible cases:

   (a) e and $-e$ are only contained in α_1.

   (b) e and $-e$ are only contained in α_2.

   (c) e and $-e$ are contained in both α_1 and α_2.

   Since both α_1 and α_2 are simple then c_{α_1} and c_{α_2} are equal to one by **Proposition 4.2.7**. Hence the three cases correspond to:

   (a) $\cup_{\Delta\in\alpha_2}\Delta \subsetneq \cup_{\Delta\in\alpha_1}\Delta$.

   (b) $\cup_{\Delta\in\alpha_1}\Delta \subsetneq \cup_{\Delta\in\alpha_2}\Delta$.

   (c) $\cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha_1}\Delta$.

   Now $\cup_{\Delta\in\alpha_1\sqcup\alpha_2}\Delta = \cup_{\Delta\in\alpha}\Delta$, so the three cases become:

   (a) $\cup_{\Delta\in\alpha_1}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   (b) $\cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   (c) $\cup_{\Delta\in\alpha_1}\Delta = \cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   Hence all three cases will thus result in $c_\alpha = 1$, since $c_{\alpha_1} = 1$ and $c_{\alpha_2} = 1$.

5. Following 4) we have three different cases for the maximum and minimum values:

   (a) $\cup_{\Delta\in\alpha_2}\Delta \subsetneq \cup_{\Delta\in\alpha_1}\Delta$, so $S_{\alpha_2} < S_{\alpha_1}$.

   (b) $\cup_{\Delta\in\alpha_1}\Delta \subsetneq \cup_{\Delta\in\alpha_2}\Delta$, so $S_{\alpha_1} < S_{\alpha_2}$.

   (c) $\cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha_1}\Delta$, so $S_{\alpha_2} = S_{\alpha_1}$.

   We also know that this corresponds to:

   (a) $\cup_{\Delta\in\alpha_1}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   (b) $\cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   (c) $\cup_{\Delta\in\alpha_1}\Delta = \cup_{\Delta\in\alpha_2}\Delta = \cup_{\Delta\in\alpha}\Delta$.

   Hence,

(a) $S_{\alpha_1} = S_\alpha$.

(b) $S_{\alpha_2} = S_\alpha$.

(c) $S_{\alpha_1} = S_{\alpha_2} = S_\alpha$.

Therefore,

$$S_\alpha = \max\{S_{\alpha_1}, S_{\alpha_2}\} = \max\{n_{\alpha_1} + L_{\alpha_1} - 1, n_{\alpha_2} + L_{\alpha_2} - 1\},$$

by **Proposition 4.2.7**. □

Proposition 4.2.30. *Let α be a symmetric multisegment then the dual multisegment $\tilde{\alpha}$ will also be symmetric.*

Proof. Let e be the maximum value of the multisegment α. Then the Mœglin-Waldspurger algorithm will construct a segment ending in e and starting with $b \leq e$ from a list of preceding segments

$$L = \{\Delta_b, \ldots, \Delta_e\}.$$

Following this the remaining multisegment is given by $\alpha - \Delta'(\alpha)$. By **Theorem 3.1.8**, we can also use the alternate Mœglin-Waldspurger algorithm on α, and by symmetry the alternate algorithm should construct the segment $[-e, -b]$, so let $-L$ denote the list of proceeding segments chosen,

$$-L = \{\Delta_{-e}, \ldots, \Delta_{-b}\} = \{-\Delta_e, \ldots, -\Delta_b\}.$$

If L and $-L$ are distinct then when the alternate algorithm is used on $\alpha - \Delta'(\alpha)$ it constructs $[-e, -b]$, and $\alpha - \Delta'(\alpha) - \Delta''(\alpha)$ will thus be symmetric, since the end is removed from each Δ_i and the start is removed from each $-\Delta_i$.

Alternatively, if L and $-L$ are not distinct, then we can consider two cases. The first is when L and $-L$ share a singleton Δ. If this is the case then by **Lemma 4.2.16** the segments in L after Δ and before Δ in $-L$ must all be singletons. This implies that there exists a string of singletons from $-e$ to e, which must be given by both $-L$ and L. Thus the segment generated will be symmetric, and the removal of the segments to create $\alpha - \Delta'(\alpha)$ will thus result in a symmetric multisegment.

Instead let us assume that the shared segment $\Delta = [b', e']$ is not a singleton. Then by **Proposition 3.1.9**, the segments before Δ in L will have length greater than or equal to Δ and after Δ will have length less than or equal to Δ. Likewise, the segments before Δ in $-L$ will have length less than or equal to Δ and after Δ will have length greater than or equal to Δ. Thus the segments before Δ in L must have the same length as Δ, otherwise we would have a contradiction since there exists a string of segments given in $-L$ have length less than or equal to Δ and satisfy the preceding property. In addition this also implies that the segments after Δ in $-L$ must have the same length as Δ. This implies that there exists a string of minimal length preceding segments, which must be given by both $-L$ and L. Following an iteration of the Mœglin-Waldspurger algorithm followed by another of the alternate Mœglin-Waldspurger algorithm, we therefore have computed the two segments $[b, e]$ and $[-e, -b]$ which are symmetric, and the remaining multisegment $\alpha - \Delta'(\alpha) - \Delta''(\alpha)$ will also be symmetric.

Therefore, given any symmetric multisegment then following either one or two iterations of the algorithm $\Delta'(\alpha)$ and $-\Delta'(\alpha)$ will be added to the dual, and the remaining multisegment will also be symmetric. Thus by a recursive argument the dual of α will be symmetric. □

We can extend this study further by studying any multisegment which is formed by the union of m simple symmetric multisegments for $m \geq 2$. Unsurprisingly the results for the higher dimensional simply follow from our study of the case when $m = 2$.

Proposition 4.2.31. *If α be any multisegment which is formed by taking the union of m simple symmetric multisegments $\alpha_1, \ldots \alpha_m$, then we can verify:*

1. *The dual of α will also be formed by taking the union of m simple symmetric multisegments.*

2. $n_{\tilde{\alpha}} = L_{\alpha_1} + \cdots + L_{\alpha_m}$.

3. $L_{\tilde{\alpha}} = \max\{n_{\alpha_1}, \ldots, n_{\alpha_m}\}$.

4. $c_\alpha = 1$.

5. $S_\alpha = \max\{S_{\alpha_1}, \ldots, S_{\alpha_m}\} = \max\{n_{\alpha_1} + L_{\alpha_1} - 1, \ldots, n_{\alpha_m} + L_{\alpha_m} - 1\}$.

Proof. 1. **Proposition 4.2.27** states that $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_2}$, and by **Proposition 4.2.7**, $\widetilde{\alpha_1}, \ldots, \widetilde{\alpha_m}$ will also be simple. Following the proof of **Proposition 4.2.27**, $\widetilde{\alpha_1}, \ldots, \widetilde{\alpha_m}$ will all be symmetric. Thus the dual of α will be formed by the union of m simple symmetric multisegments.

2. Firstly, $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}$ by **Proposition 4.2.27**, thus $n_{\tilde{\alpha}} = n_{\widetilde{\alpha_1}} + \cdots + n_{\widetilde{\alpha_m}}$. Note $\alpha_1, \ldots, \alpha_m$ are all simple thus by **Proposition 4.2.7** $n_{\widetilde{\alpha_1}} = L_{\alpha_1}, \ldots, n_{\widetilde{\alpha_m}} = L_{\alpha_m}$, so $n_{\tilde{\alpha}} = L_{\alpha_1} + \cdots + L_{\alpha_m}$.

3. $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}$ by **Proposition 4.2.27**, therefore the segment of maximum length either results from one of $\widetilde{\alpha_1}, \ldots, \widetilde{\alpha_m}$. Hence

$$L_{\tilde{\alpha}} = \max\left\{L_{\widetilde{\alpha_1}}, \ldots, L_{\widetilde{\alpha_m}}\right\} = \max\left\{n_{\alpha_1}, \ldots, n_{\alpha_m}\right\},$$

by **Proposition 4.2.7**.

4. Let us assume that e is the maximum value of α then by symmetry $-e$ will be the minimum value, and hence both $-e$ and e must be contained inside at least one α_i. So let us assume that $-e$ and e are contained inside α_j, then

$$\cup_{\Delta \in \alpha_j} \Delta = (-e, -e+1, \ldots, e-1, e) \subseteq \cup_{\Delta \in \alpha} \Delta,$$

since α_j is simple. We also know that by assumption $-e$ and e are the minimum and maximum values respectively contained inside of α, which implies that $(-e, -e+1, \ldots, e-1, e)$ is thus the maximal subset of $\cup_{\Delta \in \alpha} \Delta$. Therefore

$$\cup_{\Delta \in \alpha_j} \Delta = (-e, -e+1, \ldots, e-1, e) = \cup_{\Delta \in \alpha} \Delta,$$

so $c_\alpha = 1$ since $c_{\alpha_j} = 1$ by **Proposition 4.2.7**.

5. We already know that there exists a maximum value of α e and $-e$ will be the minimum value, by **Proposition 4.2.7** each $c_{\alpha_i} = 1$, 4) states $c_\alpha = 1$. Thus following 4) there exists at least one α_j containing both $-e$, e and it is such that

$$\cup_{\Delta \in \alpha_j} \Delta = (-e, -e+1, \ldots, e-1, e) = \cup_{\Delta \in \alpha} \Delta.$$

Therefore, $S_\alpha = S_{\alpha_j}$ for all such j. Recall that each

$$\cup_{\Delta\in\alpha_i}\Delta \subseteq \cup_{\Delta\in\alpha}\Delta,$$

thus $S_{\alpha_i} \leq S_\alpha$ for all $1 \leq i \leq m$. Therefore,

$$S_\alpha = \max\{S_{\alpha_1},\dots,S_{\alpha_m}\} = \max\{n_{\alpha_1}+L_{\alpha_1}-1,\dots,n_{\alpha_m}+L_{\alpha_m}-1\},$$

by **Proposition 4.2.7**. □

We will now present a couple of lemmas involving the Mœglin-Waldspurger algorithm and symmetric simple sub-multisegments which will then be used in the proof that the partial ordering relation is satisfied for all Langlands parameters of Arthur type.

Lemma 4.2.32. *Let α be an arbitrary multisegment containing a sub-multisegment α_1 of the form*

$$\alpha_1 = \{[-e,b],[-e+1,b+1],\dots,[-b-1,e-1],[-b,e]\}.$$

If α_1 contains both of the shortest segments containing the minimum and maximum values, $-e$ and e, of the multisegment α then it will not be possible to generate $[b,e]$ or a segment containing $b,\dots,e$ from any sub-multisegment other than α_1.

Proof. Firstly, by **Remark 4.2.24** when generating a segment for the dual we generate a sub-multisegment γ which forms an irreducible ladder multisegment. Thus γ must be such that it satisfies

$$n_\gamma + n_{\tilde{\gamma}} - c_\gamma = S_\gamma,$$

by **Lemma 4.2.19**. Now $S_\gamma \leq S_\alpha = 2e+1$, since γ is a sub-multisegment of α. Also γ is irreducible so $c_\gamma = 1$, and the segment created contains $b,\dots,e$ so $n_\gamma \geq e-b+1$. By assumption, e is the maximum value and the shortest segment ending in e has length $e-(-b)+1$ since α is both simple and symmetric, so $L_\gamma \geq e+b+1$ by **Proposition 3.1.9** which implies $n_{\tilde{\gamma}} \geq L_\gamma \geq e+b+1$ from **Lemma 4.2.2**. Using these values then we find that

$$S_\gamma = n_\gamma + n_{\tilde{\gamma}} - c_\gamma \geq (e-b+1)+(e+b+1)-1 = 2e+1.$$

Therefore $S_\gamma = 2e+1$, and $n_\gamma, n_{\tilde{\gamma}}, L_\gamma$ must all be minimal. Therefore by **Proposition 3.1.9**, every segment in γ must be of the minimal length $e+b+1$, and there must exist $e-b+1$ segments covering the values from $[-e,e]$, so the only possible formation for γ is the sub-multisegment α_1. □

Lemma 4.2.33. *Let α be an arbitrary multisegment containing a sub-multisegment α_1 of the form*

$$\alpha_1 = \{[-e,b],[-e+1,b+1],\dots,[-b-1,e-1],[-b,e]\} = \left\{\Delta^b,\Delta^{b+1},\dots,\Delta^{e-1},\Delta^e\right\}.$$

If α_1 contains both of the shortest segments containing the minimum and maximum values, $-e$ and e, of the multisegment α then removing copies of α_1 will induce an endoscopic decomposition, that is,

$$\alpha = \alpha_1 \sqcup (\alpha - \alpha_1) \text{ and } \tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{(\alpha - \alpha_1)}.$$

Proof. Firstly, it follows directly that α will be equal to the union of α_1 and $(\alpha - \alpha_1)$. So what remains is to show that the union of $\widetilde{\alpha_1}$ and $\widetilde{(\alpha - \alpha_1)}$ will be equal to $\tilde{\alpha}$ when the Mœglin-Waldspurger algorithm is used. To show this we can use **Corollary 3.2.12** which allows us to use the initial fixed set of preceding relations when carrying out the algorithm. The first segment constructed by the Mœglin-Waldspurger is $[b,e]$ using all of the ends values of the segments contained in α_1. Note for any subsequent segment that ends in e constructed by the algorithm, the segments chosen cannot be from α_1 by **Proposition 3.1.12**. Thus any segment which is constructed and ending in e must be constructed from the original segments.

If we now study the remaining integers in the segment $[-b,e]$, then for all $-b \le i \le e-1$ we know that i must end segments in the dual since there are no segments in α which initially precede $[-b,e]$. Let us assume that we have the first iteration for which an integer j originally contained in the segment Δ^k from α_1 for some $b \le k \le e-1$ such that j is chosen in a prior iteration to $j+1$ from Δ^{k+1}. Without loss of generality let us assume this happens when constructing segments with end value $i = l$ then the segment Δ^k has already been chosen in $e-l$ iterations prior to this before being chosen as Δ_j by the algorithm. Note $e-l$ iterations is the maximum possible number of iterations than any segment can have been chosen in prior to this current iteration by **Proposition 3.1.12**. Now in order for Δ_{j+1} to end in $j+1$ at this current iteration and initially precede then it must also have been chosen in exactly $e-l$ iterations prior to this. Similarly it follows that all segments chosen before Δ_j during this current iteration

$$\Delta_{j+1}, \dots, \Delta_i,$$

must also have been chosen in exactly $e-l$ previous iterations. This results in the original segment Δ corresponding to Δ_i ending, and the base value of Δ must be equal to the base value of Δ^e, since it is chosen before the segment corresponding to Δ^e (which now also ends in i) and Δ^e was initially the shortest segment containing e by assumption. Therefore Δ_i is equal to the segment corresponding to Δ^e at this current iteration, and by **Proposition 3.1.9**, Δ_i and Δ_j being the same length implies all segment chosen in between them by the algorithm must have the same length. Therefore the segments Δ_m for $j+1 \leq m \leq i$ will be equal to the segments from α_1 at this iteration, so we can instead choose these segments from α_1. The list of shortest segments preceding Δ_j will then be formed segments in α_1 since they are of minimal length, and there cannot exist a segment preceding that corresponding to Δ^b, since it did not originally precede. Hence for all $-b \leq i \leq e-1$ the segment $[i-(e-b), i]$ will be constructed exclusively from integers contained in α_1.

Therefore for all integers in Δ^e the preceding segments which are chosen will be contained in the sub-multisegment α_1, and hence the dual of α_1 is chosen independently, so

$$\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \widetilde{(\alpha - \alpha_1)}.$$

Hence $\alpha = \alpha_1 \sqcup (\alpha - \alpha_1)$ will form an endoscopic decomposition. $\square$

Remark 4.2.34. **Lemma 4.2.33** generalises the results from **Proposition 4.2.27**.

Theorem 4.2.35. *Let α be a multisegment formed by the taking the union of m simple symmetric multisegments. If β is a multisegment such that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, then $\alpha = \beta$.*

Proof. Firstly, let α be a multisegment formed by the taking the union of m simple symmetric multisegments, and let us assume that there exists a multisegment β such that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$. Then by assumption $\tilde{\alpha} \leq \tilde{\beta}$ so $r_{\tilde{\alpha},i,j} \leq r_{\tilde{\beta},i,j}$ for all i, j, where the rank $r_{\tilde{\alpha},i,j}$ denotes the number of appearances of the sequence $i, \dots, j$ in the segments of $\tilde{\alpha}$. In other words, $r_{\tilde{\alpha},i,j}$ is the number of segments $[k,l]$ contained in $\tilde{\alpha}$ such that $k \leq i$ and $j \leq l$ as discussed in the **Rank Triangle Construction algorithm** in Section 2.3. There will exist a maximum value of the multisegment denoted by e then $-e$ will be the minimum value. So when the Mœglin-Waldspurger algorithm is taken on α then it will choose the segment containing e which is shortest and denoted Δ_e. The segment Δ_e will be part of a simple symmetric multisegment α_1 which forms α, and the algorithm will hence generate a segment $[b,e]$ from this simple symmetric multisegment

$$\alpha_1 = \{[-e,b],\ldots,[-b,e]\}.$$

The segment $\Delta_e = [-b,e]$ will be the shortest segment containing e in α and by **Lemma 4.2.32**, the formation of the multisegment α_1 results in it being the only possible contributing factor to $r_{\tilde{\alpha},b,e}$, hence $r_{\tilde{\alpha},b,e}$ simply denotes the number of copies of α_1 in α.

If we study $r_{\tilde{\alpha},b,e}$ and $r_{\tilde{\beta},b,e}$, then we know that $r_{\tilde{\beta},b,e}$ must be at least $r_{\tilde{\alpha},b,e}$. In order, to have $r_{\tilde{\alpha},b,e} < r_{\tilde{\beta},b,e}$, then **Lemma 4.2.32** also implies that this would require us to create shorter segments containing e. However to do this in the formation of β, we would be required to use either union intersection or conjunction. We can immediately rule out the use of conjunction, since this only creates a longer segment. If we now look at union intersection, then the shorter segment which is created will be formed by those values which are repeated by the two segments that the action is taken on. So e must appear in both in order to be in the shorter segment, however if e appears in both then the union intersection will be equal to the shorter segment. Consequently, it is not possible to generate a shorter segment containing e in α.

Therefore $r_{\tilde{\alpha},b,e} = r_{\tilde{\beta},b,e}$ and as demonstrated in **Lemma 4.2.32** α_1 is the only possible sub-multisegment which can generate $[b,e]$. Additionally, it will not be possible to perform any actions on any of the other segments in α_1, because any operation on the segments in α_1 would change them, and could no longer be used to form $[b,e]$. Thus each copy of α_1 (α could include multiple copies) will also be sub-multisegments used to form β since it is the only possible sub-multisegment which contributes to $r_{\tilde{\beta},b,e}$. We can now use **Lemma 4.2.33** to find

$$\alpha = \alpha_1 \sqcup (\alpha - \alpha_1) \quad \text{and} \quad \beta = \alpha_1 \sqcup (\beta - \alpha_1)$$

will form endoscopic decompositions. The multisegment that remains $(\alpha - \alpha_1)$ following the removal will also be a union of simple symmetric multisegments, thus we can use a recursive argument on the maximum value e and shortest segment containing it Δ_e until we reach the case in which the multisegment is formed by a single symmetric multisegment or is empty. If we reach the case the multisegment is formed by a single symmetric multisegment, then we can use **Lemma 4.2.8** to show that this should also remain fixed. Therefore, since all m simple symmetric multisegments in α will be used as sub-multisegments in the formation of β then $\alpha = \beta$. □

Therefore we have proved that the partial ordering relation will be satisfied for ABV-packets for orbits of Arthur type. The following corollary proves the significant conjecture: *ABV-packets for orbits of Arthur type in GL_n are singletons*, which was first proposed by Cunningham et al. [4] .

Corollary 4.2.36. *ABV-packets for orbits of Arthur type are singletons and consequently, ABV-packets for orbits of Arthur type are A-packets.*

4.2.3 Further Families of Multisegments

In the previous subsection, we proved that the partial ordering relation will be satisfied for the family of multisegments of Arthur type, that is, those multisegments which are unions of simple symmetric multisegments. We now seek to broaden this family of multisegments. In Section 4.2.1, we inferred that the partial ordering relation would hold for a single ladder multisegment (**Theorem 4.2.23**) following the argument for a single simple multisegment (**Theorem 4.2.9**). One may therefore expect that the partial ordering relation will hold for a multisegment which is formed by the union of symmetric ladder multisegments. Especially since, each of the ladder multisegments can be considered as a sub-multisegment α_i which forms an endoscopic decomposition of α, and individually each will satisfy the partial ordering relation by **Theorem 4.2.23**. However as the next example will demonstrate there are a number of extra subtleties to consider for these cases.

Example 4.2.37. Let us consider the multisegment given by

$$\alpha = \{[-3,1],[-2,0],[-1,3],[0,2]\},$$

then α can be partitioned into ladder multisegments $\alpha_1 = \{[-3,1],[-1,3]\}$ and $\alpha_2 = \{[-2,0],[0,2]\}$, which form an endoscopic decomposition. However the partial ordering relation will not be satisfied since there exists the following five exceptions to the rule:

$$\beta_1 = \{[-3,3],[-2,0],[-1,1],[0,2]\}, \qquad \beta_2 = \{[-3,1],[-2,2],[-1,3],[0]\},$$
$$\beta_3 = \{[-3,1],[-2,3],[-1,0],[0,2]\}, \qquad \beta_4 = \{[-3,2],[-2,0],[-1,2],[0,1]\},$$
$$\beta_5 = \{[-3,2],[-2,3],[-1,0],[0,1]\}.$$

Therefore there exists a couple of different structures in which the partial ordering relation ultimately fails. Firstly, we can see in β_1 and β_2 that through either the union or intersection actions it is possible to generate a new segment which fits into one of the two already established ladder multisegments. Alternatively, there can exist more complex cases in which some segments remain fixed and new segments are created as shown by β_3 and β_4. Further still β_5 shows that it can be possible to generate a completely distinct set of segments that will still form two ladder multisegments.

That being said, there are a couple of cases which directly follow from the proof of **Theorem 4.2.35**.

Theorem 4.2.38. *Let α_1 be a simple symmetric multisegment, α_2 a symmetric ladder multisegment, and $\alpha = \alpha_1 \sqcup \alpha_2$. Let the maximum value e_{α_1} of α_1 be greater than or equal to the maximum value of α_2, and if it is equal then segment in α_1 containing e_{α_1} is of shorter length than the segment containing e_{α_1} in α_2. If β is a multisegment such that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, then $\alpha = \beta$.*

Proof. Firstly, let α be the union of α_1, a simple symmetric multisegment, and α_2, a symmetric ladder multisegment. Let us also impose the conditions that the maximum value e_{α_1} of α_1 be greater than or equal to the maximum value of α_2, and if it is equal then segment in α_1 containing e_{α_1} is of shorter length than the segment containing e_{α_1} in α_2. Let us assume that there exists a multisegment β such that $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$. Then by assumption $\tilde{\alpha} \leq \tilde{\beta}$ so $r_{\tilde{\alpha},i,j} \leq r_{\tilde{\beta},i,j}$ for all i, j, where the rank $r_{\tilde{\alpha},i,j}$ denotes the number of appearances of the sequence $i, \dots, j$ in the segments of $\tilde{\alpha}$. In other words, $r_{\tilde{\alpha},i,j}$ is the number of segments $[k,l]$ contained in $\tilde{\alpha}$ such that $k \leq i$ and $j \leq l$ as discussed in the **Rank Triangle Construction algorithm** in Section 2.3. There will exist a maximum value of the multisegment denoted by e_{α_1} then $-e_{\alpha_1}$ will be the minimum value. So when the Mœglin-Waldspurger algorithm is taken on α then it will choose the segment containing e_{α_1} which is shortest and denoted $\Delta_{e_{\alpha_1}}$. The segment $\Delta_{e_{\alpha_1}}$ will be part of a simple symmetric multisegment α_1 which forms α, and the algorithm will hence generate a segment $[b_{\alpha_1}, e_{\alpha_1}]$ from this simple symmetric multisegment

$$\alpha_1 = \{[-e_{\alpha_1}, b_{\alpha_1}], \ldots, [-b_{\alpha_1}, e_{\alpha_1}]\}.$$

The segment $\Delta_{e_{\alpha_1}} = [-b_{\alpha_1}, e_{\alpha_1}]$ will be the shortest segment containing e_{α_1} in α and following **Lemma 4.2.32**, the formation of the multisegment α_1 results in it being the only possible contributing factor to $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}}$, hence $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}}$ simply denotes the number of copies of α_1 in α.

If we study $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}}$ and $r_{\tilde{\beta}, b_{\alpha_1}, e_{\alpha_1}}$, then we know that $r_{\tilde{\beta}, b_{\alpha_1}, e_{\alpha_1}}$ must be at least $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}}$. In order, to have $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}} < r_{\tilde{\beta}, b_{\alpha_1}, e_{\alpha_1}}$, then **Lemma 4.2.32** also implies that this would require us to create shorter segments containing e_{α_1}. However to do this in the formation of β, we would be required to use either union intersection or conjunction. We can immediately rule out the use of conjunction, since this only creates a longer segment. If we now look at union intersection, then the shorter segment which is created will be formed by those values which are repeated by the two segments that the action is taken on. So e_{α_1} must appear in both in order to be in the shorter segment, however if e_{α_1} appears in both then the union intersection will be equal to the shorter segment. Consequently, it is not possible to generate a shorter segment containing e_{α_1} in α.

Therefore $r_{\tilde{\alpha}, b_{\alpha_1}, e_{\alpha_1}} = r_{\tilde{\beta}, b_{\alpha_1}, e_{\alpha_1}}$ and as demonstrated in **Lemma 4.2.32** α_1 is the only possible sub-multisegment which can generate $[b, e]$. Additionally, it will not be possible to perform any actions on any of the other segments in α_1, because any operation on the segments in α_1 would change them, and could no longer be used to form $[b, e]$. Thus each copy of α_1 (α could include multiple copies) will also be sub-multisegments used to form β since it is the only possible sub-multisegment which contributes to $r_{\tilde{\beta}, b_{\alpha_1}, e_{\alpha_1}}$. We can now use **Lemma 4.2.33** to find

$$\alpha = \alpha_1 \sqcup (\alpha - \alpha_1) \quad \text{and} \quad \beta = \alpha_1 \sqcup (\beta - \alpha_1)$$

will form endoscopic decompositions. The multisegment that remains $(\alpha - \alpha_1)$ following the removal will be a symmetric ladder multisegment, thus we can invoke **Theorem 4.2.23** to show that this should also remain fixed. Therefore, since both sub-multisegments α_1 and α_2 must remain fixed in α then $\alpha = \beta$. $\square$

Following on from this we can present a more generalised version of the theorem.

Theorem 4.2.39. *Let* $\alpha_1, \alpha_2, \ldots, \alpha_{m-1}$ *be a simple symmetric multisegments,* α_m *a symmetric ladder multisegment, and*

$$\alpha = \alpha_1 \sqcup \alpha_2 \sqcup \cdots \sqcup \alpha_m.$$

Let the respective maximum values $e_{\alpha_1}, \ldots, e_{\alpha_{m-1}}$ *of* $\alpha_1, \ldots, \alpha_{m-1}$ *be greater than or equal to the maximum value of* α_m*. When the maximum value of* e_{α_i} *is equal to* e_{α_m} *then the segment in* α_i *containing* e_{α_i} *must be of shorter length than the segment containing* e_{α_i} *in* α_m*. If* β *is a multisegment such that* $\alpha \leq \beta$ *and* $\tilde{\alpha} \leq \tilde{\beta}$*, then* $\alpha = \beta$.

Proof. This proof follows directly from the proof of **Theorem 4.2.38**. However, we instead need to recursively fix each of the simple symmetric multisegments $\alpha_1, \alpha_2, \ldots, \alpha_{m-1}$, as demonstrated in the proof of **Theorem 4.2.35** using **Lemma 4.2.32** and **Lemma 4.2.33**. □

However, these theorems will not generally be satisfied following the removal of conditions on the maximum values and shortest segment, as will be demonstrated in the following example.

Example 4.2.40. Let α_1 be a simple symmetric multisegment and α_2 be a symmetric ladder multisegment. If we ignore the assumptions and set

$$\alpha_1 = \{[-1,0],[0,1]\} \text{ and } \alpha_1 = \{[-2,0],[0,2]\},$$

then

$$\widetilde{\alpha_1 \sqcup \alpha_2} = \{[-2],[-1,0],[-1,0],[0,1],[0,1],[2]\}.$$

If we now set

$$\beta = \{[-2,0],[-1,1],[0,2],[0]\},$$

then β is self-dual, so $\alpha \leq \beta$ and $\tilde{\alpha} \leq \tilde{\beta}$, but $\alpha \neq \beta$. Thus we have a counter example.

Therefore it has become increasingly difficult to now define further families of multisegments for which the partial ordering relation will always be satisfied for.

4.3 Constructing Counter Examples to the Partial Ordering Relation

Up unto this point in Chapter 4, we have attempted to describe different families of multisegments for which the partial ordering relation will always be satisfied. Throughout this section we will instead study the opposite of the question, that is, we will search for the families of multisegment for which the partial ordering relation will not be satisfied. To do this we will fix the top row of a rank triangle and then seek to study all of the counter examples which can then be formed. A key factor that we can use in this categorisation is the relationship between counter examples and endoscopic decompositions.

Proposition 4.3.1. *Let α be any multisegment and $\alpha_1, \ldots \alpha_m$ be sub-multisegments of α which form an endoscopic decomposition such that $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}$. If any of the sub-multisegments α_i are such that there exists a β_i for which $\alpha_i \leq \beta_i$ and $\widetilde{\alpha_i} \leq \widetilde{\beta_i}$ but $\alpha_i \neq \beta_i$, then there also exists β for which $\alpha \leq \beta$ and $\widetilde{\alpha} \leq \widetilde{\beta}$ but $\alpha \neq \beta$.*

Proof. For this proof we will invoke the idea used in Chapter 3 that the Mœglin-Waldspurger algorithm can be carried out by maximum flows in a network. Let α be any multisegment and $\alpha_1, \ldots \alpha_m$ be sub-multisegments of α which forms an endoscopic decomposition such that $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}$. Let us assume that there exists at least one sub-multisegments α_i for which there is a β_i such that $\alpha_i \leq \beta_i$ and $\widetilde{\alpha_i} \leq \widetilde{\beta_i}$. Then by construction the maximum flow on at least one of the iterative steps for both the original network and the dual of α_i must increase following the creation of β_i. If we now impose the same actions that creates β_i on the whole multisegment. Then the new multisegment generated will contain each of the subgraphs associated to the α_j's for $j \neq i$ and β_i. Since we know that sum of the flow associated to each of the subgraphs at each iterative step is greater than or equal to the flow at the same step for α, and at at least one iterative step the flow is actually greater than. Then this guarantees that following the addition of extra edges into the network in β that the flow at each iterative step is greater than or equal, and at at least one iterative step the flow is actually greater, since the addition of extra edges can only increase the flow. Hence $\alpha \leq \beta$. Likewise, the same will be true for the networks associated to both $\tilde{\alpha}$ and $\tilde{\beta}$. Hence $\alpha \leq \beta$ and $\widetilde{\alpha} \leq \widetilde{\beta}$ but $\alpha \neq \beta$. □

One may expect us now to be able to simply study the smaller examples and then be able to categorise all possible counter examples from these cases. However, it is possible to have an endoscopic decomposition $\alpha = \alpha_1 \cup \alpha_2$ such that α_1 and α_2 satisfy the partial ordering relation individually but α does not. This will be demonstrated in the following example:

Example 4.3.2. Let

$$\alpha = \{[-2],[-1],[-1],[0],[0,1]\}.$$

Then set

$$\alpha_1 = \{[-2],[-1],[0]\} \text{ and } \alpha_2 = \{[-1],[0,1]\},$$

then α_1 is simple and α_2 is a ladder multisegment, so each will individually satisfy the partial ordering relation. However, if we take the conjunction action on the segments $[-2],[-1]$ and $[-1],[0]$ in α, then we form

$$\beta = \{[-2,-1],[-1,0],[0,1]\},$$

which is such that $\alpha \leq \beta$, $\tilde{\alpha} \leq \tilde{\beta}$ but $\alpha \neq \beta$.

We therefore want to classify the smallest sub-multisegments for which the partial ordering relation is not satisfied.

Definition 4.3.3. Let α be a multisegment. We say α is an *indecomposable counter example* if α does not satisfy the partial ordering relation, and there does not exist a sub-multisegment α_1 of α which also does not satisfy the partial ordering relation and is such that

$$\tilde{\alpha} = \widetilde{\alpha_1} \sqcup (\widetilde{\alpha - \alpha_1}).$$

Our goal for this section will be to provide some first steps toward the classification of all indecomposable counter examples based upon number of eigenspaces (the length of the quiver). Following this we can then invoke **Proposition 4.3.1** to construct counter examples of the partial ordering relation.

4.3.1 Type A$_3$ Quivers

Let us consider the first non-trivial case in which we have 3 eigenspaces and hence a type A_3 quiver. We seek to find all indecomposable counter examples for length 3 quivers. Let α be the multisegment associated to a length 3 quiver, the the rank triangle associated to α will be of the form:

$$\begin{array}{ccccc} r_{-1,-1} & & r_{0,0} & & r_{1,1} \\ & r_{-1,0} & & r_{0,1} & \\ & & r_{-1,1} & & \end{array}.$$

We will assume that the values of the top row increase from outside to in to avoid considering a large amount of cases for which there is no interaction between outside elements. There also exists a dual rank triangle

$$\begin{array}{ccccc} \tilde{r}_{-1,-1} & & \tilde{r}_{0,0} & & \tilde{r}_{1,1} \\ & \tilde{r}_{-1,0} & & \tilde{r}_{0,1} & \\ & & \tilde{r}_{-1,1} & & \end{array},$$

which by construction will have an identical top row since the multiplicity of elements must remain the same. Recall that the partial ordering of a multisegment can be expressed in terms of ranks in the rank triangle by **Corollary 2.5.7**. Therefore, we can individually look at the cases in which the values on the rank triangle associated to $\tilde{\alpha}$ can be increased by actions taken on α to look at how counter examples can arise in this dimension.

Firstly, if we want to increase the value $\tilde{r}_{-1,1}$ then in β we need to create an extra set of singletons $[-1],[0],[1]$, since these elements are the only possible way of forming $[-1,1]$ in $\tilde{\beta}$. Note $[-1]$ and $[1]$ must already be present in α since they are unable to be generated by any action on segments. What remains is to generate $[0]$ to match with the copies of $[-1]$ and $[1]$, since it must not already be present in α otherwise this would not be an additional copy of the singleton. The only possibility for this is taking union intersection on $[-1,0]$ and $[0,1]$, so $\{[-1,0],[-1],[0,1],[1]\} \in \alpha, \tilde{\alpha}$ in order to increase $\tilde{r}_{-1,1}$.

Instead if we now look to increase $\tilde{r}_{0,1}$ then there are two possible segments which contribute to this value $[-1,1]$ and $[0,1]$. We have already discussed the way to create $[-1,1]$, thus we now need to consider creating $[0,1]$. There are three possibilities for this:

1. [0], [1] : [1] must already be present in α, and recall [0] must be generated using union intersection on $[-1,0]$ and $[0,1]$ which would result in decreasing $\tilde{r}_{-1,0}$.

2. $[-1,0]$, [1]: [1] must already be present in α, and $[-1,0]$ must be generated using conjunction on $[-1]$ and [0] which would result in decreasing $\tilde{r}_{-1,1}$.

3. $[-1,0]$, $[0,1]$: this can be generated by one of two ways either $[0,1]$ is present and conjunction is taken on $[-1]$ and [0], or, either $[-1,0]$ is present and conjunction is taken on [0] and [1].

Therefore 3. is the only possible case to generate a new copy of $[0,1]$. Note the dual formation ($[-1,0]$ and conjunction taken on [0] and [1]) will increase $\tilde{r}_{-1,0}$.

However, these conditions are necessary rather than sufficient, since for the top row (3 3 3) there exists

$$\alpha = \{[-1],[-1,0],[-1,1],[0],[1],[1]\},$$

which satisfies the partial ordering conditions for all possible β. However, $\alpha_1 = \{[-1,0],[0],[1]\}$ is contained inside α and $\widetilde{\alpha_1} = \{[-1],[0],[0,1]\}$ is also contained in $\tilde{\alpha}$. This results from the fact that $\tilde{\alpha} \neq \widetilde{\alpha_1} \sqcup (\widetilde{\alpha - \alpha_1})$.

Definition 4.3.4. Let α be a multisegment. We say that $\alpha_1 \sqcup \cdots \sqcup \alpha_m$ forms a *complete endoscopic decomposition* of α if the following conditions are satiesfied:

1. $\tilde{\alpha} = \widetilde{\alpha_1} \sqcup \cdots \sqcup \widetilde{\alpha_m}$.

2. For each α_i there does not exist an endoscopic decomposition

$$\alpha_i = \alpha_{i_1} \sqcup \cdots \sqcup \alpha_{i_n} \quad \text{and} \quad \widetilde{\alpha_i} = \widetilde{\alpha_{i_1}} \sqcup \cdots \sqcup \widetilde{\alpha_{i_n}}$$

   such that $n \geq 2$.

For the length 3 case, we can present a greedy algorithm for constructing the complete endoscopic decomposition with C_α components given by:

Algorithm : Complete Endoscopic Decomposition for a Type $\mathbf{A_3}$ Quiver Given a quiver of length 3 and associated multisegment α, then to construct the maximal endoscopic decomposition we spit the multisegment up by partitioning into sub-multisegments in the following order:

1. $\{[-1],[0],[1]\}$,
2. $\{[0],[1]\}$,
3. $\{[-1,0],[0,1]\}$,
4. $\{[-1,0],[1]\}$,
5. $\{[1]\}$,
6. $\{[-1],[0,1]\}$,
7. $\{[0,1]\}$,
8. $\{[-1],[0]\}$,
9. $\{[0]\}$,
10. $\{[-1,0]\}$,
11. $\{[-1]\}$,
12. $\{[-1,1]\}$

Proposition 4.3.5. *The* Algorithm : Complete Endoscopic Decomposition for a Type $\mathbf{A_3}$ Quiver *will always construct a complete endoscopic decomposition.*

Proof. Firstly, let us carry out the Mœglin-Waldspurger algorithm using the fixed preceding relations by **Corollary 3.2.12**. 1. follows by **Lemma 4.2.32** and **Lemma 4.2.33**, and 2. as an immediate consequence, since $[0]$ is the shortest segment preceding $[1]$ and since there no longer exists a complete copy of singletons there can be no preceding segments. The next one is slightly counterintuitive since one may expect both 4. and 5. to be chosen first since $[1]$ is shorter than $[0,1]$. However, if we consider each copy of $[-1,0]$ then the only segments which it precedes are $[1]$ and $[0,1]$. If $[1]$ is chosen with $[-1,0]$ by the algorithm, then on a subsequent iteration $[-1,0]$ can only be formed from the -1 in $[-1,0]$ if there exists a 0 from $[0,1]$. Similarly, $[0,1]$ can only be formed from the 1 in $[0,1]$ if there exists a 0 from $[-1,0]$, otherwise it would simply be a singleton $[1]$. Therefore we can simply partition each copy of $\{[-1,0],[0,1]\}$, since the remaining $[1]$'s will still form the singletons or be available to be matched with $[-1,0]$'s. Therefore 3. will form an endoscopic decomposition and should be chosen next. $[1]$ will still be the shortest possible segment so is chosen next then either there exists $[-1,0]$ which precedes or doesn't so it forms a singleton. Similarly, $[0,1]$ will still be the shortest possible segment so chosen next then either there exists $[-1]$ which precedes or doesn't so it forms a singletons. Only segment which will precede $[0]$ is $[-1]$ therefore 8. is next. 9., 10., 11., and 12. will finally follow as there are no preceding segments in each case thus must simply generate singletons. $\square$

Therefore we can combine this with our study of the various counter examples to obtain:

Theorem 4.3.6. *Given any length 3 quiver with associated multisegment α then α does not satisfy the partial ordering relation if any of the following sub-multisegments are part of the complete endoscopic decomposition:*

1. $\alpha_1 = \{[-1,0],[-1],[0,1],[1]\}$*, that is, 3., 5., 11. all appear in the complete endoscopic decomposition;*

2. $\alpha_2 = \{[-1,0],[0],[1]\}$*, that is, 2., 10. both appear in the complete endoscopic decomposition;*

3. $\alpha_3 = \{[-1],[0],[0,1]\}$*, that is, 7., 8. both appear in the complete endoscopic decomposition.*

Proof. The proof follows from the argument above regarding the fact that following actions on these sub-multisegments values of the rank triangles will increase, and hence if the individual sub-multisegments appear in the complete endoscopic decomposition then their union will also be an endoscopic decomposition. So it follows by **Proposition 4.3.1** that α will not follow the partial ordering condition in any of these cases. □

One could conjecture that the method outlined in **Theorem 4.3.6** is the only possible ways to generate counter examples for length 3 quivers, however we have found this extremely hard to prove.

4.3.2 Higher Length Quivers

The eloquence of the case in which the quivers are of length 3 is however not reflected in the study as we increase the length of the quiver. This follows from the fact that it becomes increasingly difficult to define an explicit method for calculating a complete endoscopic decomposition. Further, the possible actions which can be taken grow exponentially and completely categorising them becomes a much more intricate process. We will instead use a numerical based argument for each length of the quiver and number of segments n to study the number of indecomposable counter examples. See Appendix A for a *Python* code implementation for finding the number of indecomposable counter examples. The following table of values is a snapshot of the results obtained whilst running the code for four weeks. The number of indecomposable counter examples remains stable for the lengths three and four for up to twenty segments. For quivers of length six finding the number of indecomposable counter examples becomes computational unfeasible, since the case when the number of segments considered was eight required over three days to complete.

Table 4.1: Number of Indecomposable Counter Examples for up to n Segments for each Length.

		Length of Quiver			
		3	**4**	**5**	**6**
Number of Segments, n	**3**	2	14	56	168
	4	3	33	197	837
	5	3	43	378	2325
	6	3	56	646	5303
	7	3	60	891	10082
	8	3	60	1131	17583
	9	3	60	1316	
	10	3	60	1420	
	11	3	60	1450	
	12	3	60	1462	
	13	3	60	1462	
	14	3	60	1462	
	15	3	60	1462	

Note there has to be more than 2 segments to create an indecomposable counter example. This follows from **Proposition 2.5.4** that the multisegment only changes if the two segments that the action is taken on precede one another, and that either 1 segment or 2 segments in which one is contained inside the other will be created thus there will be preceding relations following the action. Hence we begin our study with at least 3 segments in the multisegment and have the following proposition:

Proposition 4.3.7. *Any indecomposable counter example will not include the* full segment*, that is, the segment* $[b,e]$ *such that b is the minimum and e is the maximum value for the length of the quiver.*

Proof. Let us first assume that the full segment is contained in an indecomposable counter example. Note there are no possible segments contained in this decomposition which can precede the full segment. **Proposition 2.5.4** states that no action can be taken on the full segment. Similarly, if we use the network theoretic approach then the vertices associated to the full segment will construct a disconnected subnetwork, since there are no preceding relations even after any actions. Therefore we have a contradiction since there will exist an indecomposable counter example which excludes the full segment and hence we didn't originally have an indecomposable counter example. □

An immediate consequence is that this will also be true for the dual of the full segment.

Corollary 4.3.8. *Any indecomposable counter example will not include the string of consecutive singletons from the minimum value to the maximum.*

For the lengths 3, 4 and 5 of quivers, the number of indecomposable counter examples appears to converge at some n. This leads us to establish some conjectures and a number of corollaries which would follow from these conjectures.

Conjecture 4.3.9. In any indecomposable counter example the maximum number of times the same segment can appear in the counter example is the length of the quiver minus 2.

This is based on the empirical data of the individual lists of indecomposable counter example for the 3, 4 and 5 length quivers.

Conjecture 4.3.10. There exists a value n for which the number of indecomposable counter example converges.

A proof of this conjecture would be a significant result since it would allow us to be able to categorise every counter example to the partial ordering condition.

Corollary to Conjecture 4.3.11. There exists a maximal size of an indecomposable counter example for each length of quiver.

This would follow from conjecture that there must exist a smallest value n for which every endoscopic counter example is formed.

Corollary to Conjecture 4.3.12. There is a finite number of indecomposable counter example for each length of quiver.

This would follow from conjecture that the number of endoscopic counter examples converges.

Corollary to Conjecture 4.3.13. Every counter example for a given length of quiver must therefore arise from one of the indecomposable counter example.

If there was to exist a counter example which did not arise from an indecomposable counter example then by definition this would also be an indecomposable counter example, so would contradict the fact that it converges.

We present a heuristic argument to justify the **Conjecture 4.3.10**. If we fix the length of quiver d then there exists a maximum of d elements in any segment. Similarly, there exists a finite number of possible segments and hence preceding relations. By **Proposition 2.5.4**, the multisegment only changes when actions are taken on preceding segments, therefore given that there exists a finite number of possible preceding relations then there is also a finite number of actions. Further the network description is highly structured given that the vertices associated to i can only send flow to vertices $i-1$ and receive flow from $i+1$. Thus the process of increasing flow through the network at at least one iterative stage is very intricate, since it is very easy to decrease the flow given that the actions create a longer segment at each stage. In addition to this, we can define a number of different bounds for the maximum flows based on the number of elements, and edges between the vertices associated to two consecutive integers. Individual segments will also provide bounds on the flow. For example, any segment containing the maximum value can only get longer, therefore for each such segment $\Delta = [b, e_\alpha]$ then the furthest the flow from any vertex associated to an integer of Δ can go is $d - e_\alpha + b$. Thus we will be able to bound $\tilde{r}_{(i,e_\alpha)}$ for all i, and by symmetry we will be able to do the same for the minimum value. Therefore given this highly structured network then it is very feasible to believe that every value could therefore be bounded and hence it be shown that convergence for n will naturally follow using this network theoretic approach.

Chapter 5 : Future Work

This thesis studied the **Open Orbit Conjecture 2.4.7** for specific families of Langlands parameters, mainly those of Arthur type. It also provided a way to generate multisegments which violate the partial ordering relation through endoscopic decomposition. To continue this study, the areas which require further exploration are:

1. The **Open Orbit Conjecture 2.4.7** remains an open problem, since we found no counter examples throughout our study.
2. Refine the definition of ladder multisegments further to find new families of Langlands parameters which can be described using numerical invariants.
3. Study whether there exists a maximal size indecomposable counter example for a given length of quiver.
4. Determine whether there exists a finite number of indecomposable counter examples which will derive every possible violation of the partial ordering relation for a given length of quiver.
5. Discover if there exists an explicit method for generating the maximal endoscopic decomposition of C_α components for every length of quiver.

Bibliography

[1] J. Adams, D. Barbasch, and D. A. Vogan Jr., *The Langlands Classification and Irreducible Characters for Real Reductive Groups*, vol. 104 of Progress in Mathematics, Springer Science & Business Media, 2012.

[2] J. Arthur, *Unipotent Automorphic Representations: Conjectures*, no. 171-172 in Astérisque, Société Mathématique de France, 1989.

[3] E. T. Bell, *Exponential Polynomials*, Annals of Mathematics, 35 (1934), pp. 258–277.

[4] C. Cunningham, A. Fiori, and N. Kitt, *Appearance of the Kashiwara-Saito Singularity in the Representation Theory of p-adic GL_{16}*, arXiv preprint arXiv:2103.04538, (2021).

[5] C. Cunningham, A. Fiori, A. Moussaoui, J. Mracek, and B. Xu, *Arthur Packets for p-adic Groups by Way of Microlocal Vanishing Cycles of Perverse Sheaves, with Examples*, no. 1353, Memoirs of the American Mathematical Soc., 2022.

[6] H. Derksen and J. Weyman, *An Introduction to Quiver Representations*, vol. 184 of Graduate Studies in Mathematics, American Mathematical Soc., 2017.

[7] S. Evens and I. Mirković, *Fourier Transform and the Iwahori-Matsumoto Involution*, Duke Mathematical Journal, 86 (1997), pp. 435–464.

[8] E. Frenkel, *Commentary on "An Elementary Introduction to the Langlands Program" by Stephen Gelbart*, Bulletin of the American Mathematical Soc., 48 (2011), pp. 513–515.

[9] H. Knight and A. Zelevinskii, *Representations of Quivers of Type A and the Multisegment Duality*, Advances in Mathematics, 117 (1996), pp. 273–293.

[10] R. P. Langlands, *Letter to André Weil*. Available Online at publications.ias.edu, 1967.

[11] ——, *Problems in the Theory of Automorphic Forms to Salomon Bochner in Gratitude*, in Lectures in Modern Analysis and Applications III, C. T. Taam, ed., vol. 170 of Lecture Notes in Mathematics, Springer, 1970, pp. 18–61.

[12] E. Lapid and A. Mínguez, *On a Determinantal Formula of Tadić*, American Journal of Mathematics, 136 (2014), pp. 111–142.

[13] C. Mœglin and J.-L. Waldspurger, *Sur l'Involution de Zelevinskii*, Journal für die Reine und Angewandte Mathematik, 372 (1986), pp. 136–177.

[14] J. Mracek, *Applications of Algebraic Microlocal Analysis in Symplectic Geometry and Representation Theory*, PhD thesis, University of Toronto (Canada), 2017.

[15] S. Poljak, *Maximum Rank of Powers of a Matrix of a Given Pattern*, Proceedings of the American Mathematical Soc., 106 (1989), pp. 1137–1144.

[16] V. S. Pyasetskii, *Linear Lie Groups Acting with Finitely Many Orbits*, Functional Analysis and Its Applications, 9 (1975), pp. 351–353.

[17] D. A. VOGAN JR., *"The Local Langlands Conjecture"*, in Representation Theory of Groups and Algebras, vol. 145 of Contemp. Math., American Mathematical Soc., 1993, pp. 305–379.

[18] A. WILES, *Modular Elliptic Curves and Fermat's Last Theorem*, Annals of Mathematics, 141 (1995), pp. 443–551.

[19] A. ZELEVINSKII, *p-adic Analog of the Kazhdan-Lusztig Hypothesis*, Functional Analysis and Its Applications, 15 (1981), pp. 83–92.

Appendix A : Python Codes

The following python code implements a method to find the dual using the Mœglin-Waldspurger and the *Network Computation of the Dual* for the example found in **Section 3.3** [to run through any arbitrary example update the element of V]. The functions which are used to implement the initial code follow the output.

```
import numpy as np
import math
#Initial Element of V
A = np.array ([[0,0], [1,0], [0,0],[0,1]])
B = np.array ([[1,0,0,0], [0,1,0,0], [0,0,1,0],[0,0,0,0]])
C = np.array ([[1,0,0,0], [0,1,0,0], [0,0,0,0],[0,0,0,1]])
D = np.array ([[1,0,0,0], [0,0,1,0]])
mat=(A,B,C,D)

#Converts the Element of V into the Rank Triangle
RT=MatrixtoRT(mat)

#Computes the Multisegment Triangle from the Rank Triangle
B=RTtoMT(RT)
B1=RTtoMT(RT)
print('Multisegment Triangle', B)

#Computes Dual Multisegment via Moeglin-Waldspurger
A=MW(B)
print('Dual Multisegment',A)
Q=MStoMT(A)
print('Dual Multisegment Triangle',Q)
P=MTtoRT(Q)
print('Dual Rank Triangle',P)

#Create Dual Rank Triangle using Max Flows through Network
DT=[]
for x in range(0, len(B1)):
    ds=[]
    for y in range(0, len(B1[x])):
        C=Graph(B1, y+1, y+1+x)
        g = Graphs(C)
        max=g.ford_fulkerson()
        ds.append(max)
    DT.append(ds)
print('Dual Rank Triangle via Network',DT)

#Converts to Multisegment Triangle
print('Dual Multisegment Triangle via Network',RTtoMT(DT))
```

The output for the code is as follows:

```
Rank Triangle [[2, 4, 4, 4, 2], [2, 3, 3, 2], [1, 2, 1], [1, 1], [0]]
Multisegment Triangle [[0, 0, 0, 0, 0], [1, 1, 1, 1], [0, 0, 0], [1, 1], [0]]
Dual Multisegment [[5, 4, 3, 2], [5, 4], [4, 3, 2, 1], [4, 3], [3, 2], [2, 1]]
Dual Multisegment Triangle [[0, 0, 0, 0, 0], [1, 1, 1, 1], [0, 0, 0], [1, 1], [0
                                        ]]
Dual Rank Triangle [[2, 4, 4, 4, 2], [2, 3, 3, 2], [1, 2, 1], [1, 1], [0]]
Dual Rank Triangle via Network [[2, 4, 4, 4, 2], [2, 3, 3, 2], [1, 2, 1], [1, 1]
                                        , [0]]
Dual Multisegment Triangle via Network [[0, 0, 0, 0, 0], [1, 1, 1, 1], [0, 0, 0]
                                        , [1, 1], [0]]
```

Unsurprisingly, the output mirrors the values found in the example in **Section 3.3**.

```
#Function 1: Converts the Element of V to the Rank Triangle as demonstrated in
                                        Section 3.3
def MatrixtoRT(Q):
    L=len(Q)
    RT=[]
    R=[]
    for n in range(0,L):
        R.append(len(Q[n][0]))
    R.append(len(Q[L-1]))
    RT.append(R)
    for n in range(0,L):
        R=[]
        for m in range(0,L-n):
            M=mat[m]
            for r in range(1,n+1):
                M=np.dot (mat[m+r],M)
            R.append(np.linalg.matrix_rank(M))
        RT.append(R)
    return(RT)
```

```
#Function 2: Converts Rank to Multisegment Triangle using Proposition 2.3.3.
def RTtoMT(A):
    L=len(A)
    MT=[]
    for n in range(0,L):
        R=A[n]
        MTR=[]
        for m in range(0,L-n):
            if n==L-1:
                M= R[m]
            else :
                if m==0:
                    S=A[n+1]
                    M= R[m]-S[m]
                elif m==L-n-1:
                    S=A[n+1]
                    M= R[m]-S[m-1]
                else :
                    S=A[n+1]
                    T=A[n+2]
                    M=R[m]-S[m]-S[m-1]+T[m-1]
            MTR.append(M)
        MT.append(MTR)
    return MT
```

```
#Function 3: Computes the Moeglin-Waldspurger Alg. on Multisegment Triangle
                                                found in Section 3.1
def MW(A):
    Dual=[]
    L=len(A)
    m=L-1
    while m>-1:
        q=0
        while q<m+1 :
            for n in range(0,m+1):
                R=A[n]
                if R[m-n]>0:
                    D=[m+1]
                    O=[[m,n]]
                    b=m-n
                    m1=m-1
                    p=-1
                    while p<m1:
                        for n1 in range (0, m1 +1):
                            R=A[n1]
                            if (R[m1 - n1]>0 and b>m1-n1 and b<=m1+1):
                                O.append([m1,n1])
                                D.append(m1+1)
                                b=m1-n1
                                m1=m1-1
                                p=-1
                                break
                            else:
                                p=p+1
                    Dual.append(D)
                    I=len(O)
                    for n in range(0,I):
                        M=O[n][0]
                        N=O[n][1]
                        R=A[N]
                        R[M-N]=R[M-N]-1
                        A[N]=R
                        if N>0:
                            S=A[N-1]
                            S[M-N]=S[M-N]+1
                            A[N-1]=S
                    q=0
                    break
                else:
                    q=q+1
        m=m-1
    return Dual
```

```
#Function 4: Converts Multisegments to Multisegment Triangle
def MStoMT(M):
    S=[]
    for x in M:
        S.append(max(x))
    Max=max(S)
    S=[]
    for x in M:
        S.append(min(x))
    Min=min(S)
    MT=[]
    for x in range(0, Max-Min+1):
        mt=[]
        for y in range(0, Max-Min-x+1):
            mt.append(0)
        MT.append(mt)
    for x in M:
        b=min(x)
        e=max(x)
        MT[e-b][b-1]=MT[e-b][b-1]+1
    return MT
```

```
#Function 5: Constructs the Network constructed in Subsection 3.2.1
def Graph(A,b,e):
    S=[]
    V=[]
    L=len(A)
    q=1
    for n in range(0,L):
        for m in range(0,n+1):
            R=A[m]
            M=R[L-n-1]
            for x in range(M):
                S.append([L-n,L+m-n,q])
                q+=1
    for x in range(L):
        for y in S:
            if y[0] <= L-x <= y[1]:
                V.append([L-x,y[0],y[1],y[2]])
    LV=len(V)
    F=np.zeros((2*LV+2,2*LV+2))
    for x in range(1, LV+1):
        for y in range(1, LV+1):
            if x != y and V[x-1][0] -1 == V[y-1][0] and V[x-1][1]>V[y-1][1] and
                                                        V[x-1][2]>V[y-1][2]:
                F[x+LV,y]=1
        F[x,x+LV]=1
    for x in range(1, LV+1):
        if V[x-1][0]==e:
            F[0,x]=1
    for x in range(1, LV+1):
        if V[x-1][0]==b:
            F[x+LV,2*LV+1]=1
    return F
```

```python
#Function 6: Creates Multisegments using the Rank Triangle Construction Alg.
def MTtoRT(MT):
    L=len(MT)
    RT=[]
    for x in range(0,L):
        rt=[]
        for y in range(0,L-x):
            rt.append(0)
        RT.append(rt)
    for x in range(0,L):
        for y in range(0,L-x):
            n=MT[x][y]
            for X in range(0,x+1):
                for Y in range(y,x+y-X+1):
                    RT[X][Y]=RT[X][Y]+n
    return(RT)
```

```python
# Function 7: Computes the Ford-Fulkerson Algorithm outlined in Subsection 3.2.2
class Graphs:
    def __init__(self, graph):
        self.graph = graph
        self. ROW = len(graph)
    # Using BFS as a searching algorithm
    def searching_algo_BFS(self, s, t, parent):
        visited = [False] * (self.ROW)
        queue = []
        queue.append(s)
        visited[s] = True
        while queue:
            u = queue.pop(0)
            for ind, val in enumerate(self.graph[u]):
                if visited[ind] == False and val > 0:
                    queue.append(ind)
                    visited[ind] = True
                    parent[ind] = u
        return True if visited[t] else False
    # Applying Ford-Fulkerson algorithm
    def ford_fulkerson(self):
        sink=self.ROW -1
        source=0
        parent = [-1] * (self.ROW)
        max_flow = 0
        while self.searching_algo_BFS(source, sink, parent):
            path_flow = float("Inf")
            s = sink
            while(s != source):
                path_flow = min(path_flow, self.graph[parent[s]][s])
                s = parent[s]
            # Adding the path flows
            max_flow += path_flow
            # Updating the residual values of edges
            v = sink
            while(v != source):
                u = parent[v]
                self.graph[u][v] -= path_flow
                self.graph[v][u] += path_flow
                v = parent[v]
        return max_flow
```

The following functions can be used to computational verify that the partial ordering relation will be satisfied in both **Theorem 4.2.35** and **Theorem 4.2.38** for each multisegment A. Function 8 generates all of the possible multisegments which can be generated by the actions described in **Proposition 2.5.2**, and then Function 9 checks to see if $\tilde{A} \leq \tilde{B}$ is satisfied. Finally, Code 10 outputs a .txt file containing any counter examples cases in which $\tilde{A} \leq \tilde{B}$ and $A \neq B$.

```
# Function 8: Finds all Rank Triangles formed by actions such that A <= B
import itertools
def all_multisegments(A):
    AllRT=[[A[0]]]
    for x in range(1,len(A)):
        AllRTs=[]
        for y in AllRT:
            Numbers=[]
            for z in range(0,len(A)-x):
                Numbers.append(list(range(0,min(y[x-1][z],y[x-1][z+1])+1)))
            for element in itertools.product(*Numbers):
                if (all( int(A[x][v]) <= int(element[v]) for v in range(0,len(
                                                        element)))):
                        Status=True
                        if x==1:
                            AllRTs.append([*y, list(element)])
                            Status=False
                        for t in range(0,len(element)):
                            if int(y[x-1][t]) + int(y[x-1][t+1]) > int(element[t
                                                                    ])+int(y[x-2
                                                                    ][t+1]):
                                Status=False
                        if Status==True:
                            AllRTs.append([*y, list(element)])
        AllRT=list(AllRTs)
    return AllRT
```

```
# Function 9: Checks if Dual A <= Dual B
def DualAlesseqDualB(DualA,DualB):
    count1=0
    count2=0
    for x in range(0, len(DualA)):
        for y in range(0,len(DualA[x])):
            if DualA[x][y]<=DualB[x][y]:
                count1=count1 +1
            count2=count2+1
    if count1 == count2:
        print('Dual(A)<=Dual(B)')
```

```
# Code 10: For any multisegment A condition A <= B and Dual A <= Dual B implies
                                        A=B is checked.
Output = open("OutputManySimple.txt", "w")
AllDRT2s=all_multisegments2(RT1)
for pi in AllDRT2s:
    RT2=list(pi)
    P2=RTtoMT(RT2)
    Dual2=MW(P2)
    Num2=len(Dual2)
    DMT2=MStoMT(Dual2)
    DRT2=MTtoRT(DMT2)
    if DualAlesseqDualB(DRT1, DRT2)==True:
        if RT1 != RT2:
              Output.write('Counter Number:'+'Iteration:'+'n1'+str(n1)+'a1'
                            +str(a1)+'\n'+str(Num1)+',' +str(Num2)+'\n'+str(RT1)
                            +'\n'+str(RT2)+'\n'+str(DRT1)+'\n'+str(DRT2)+'\n\n')
            Output.flush()
```

We want to check the partial ordering relation for different families of multisegments for which we will require individual codes to construct them. In each of the following cases Code 10, which checks the partial ordering relation, will be inserted where labelled. The first case we considered was those multisegments generated by combining multiple simple symmetric multisegments given in Code 11a, and then we considered a random symmetric multisegment given in Code 11b.

```
# Code 11a: Rigorously Constructs all Multisegments of Arthur Type.
maxnumseg=10
maxlenseg=5
MaxNumberSym=5
N=[1]
for Number in range(2,MaxNumberSym+1):
    N=[*[1]*(Number-1),1]
    while N[len(N)-1]<maxnumseg:
        while len(set(N)) > 1:
            Status=False
            for z in range(0, len(N)-1):
                if N[z]<N[z+1] and Status==False:
                    N[z]=N[z]+1
                    Status = True
                    break
            A=[*[1]*(Number-1),1]
            while A[len(A)-1]<maxlenseg:
                Ni=[]
                Ai=[]
                Li=[]
                s=0
                for x in range(Number):
                    if (N[x] % 2) == 0:
                        l1=2*A[x]
                    else:
                        l1=2*A[x]-1
                    s=max(N[x]+l1, s)
                    Ni.append(N[x])
                    Ai.append(A[x])
                    Li.append(l1)
                M=[]
                for x in range(Number):
                    si=int(s)
                    s1=int((si-Ni[x]-Li[x])/2)
                    for y in range(Ni[x]):
                        S=[]
                        for z in range(s1+1,s1+1+Li[x]):
                            S.append(z)
                        s1=s1+1
                        M.append(S)
                M1=list(M)
                Num1a=len(M)
                M3=list(M)
                P=MStoMT(M1)
                Q=MStoMT(M3)
                Dual1=MW(P)
                Num1=len(Dual1)
                RT1=MTtoRT(Q)
                DMT1=MStoMT(Dual1)
                DRT1=MTtoRT(DMT1)
                # Code 10 goes here
```

```
# Code 11b: Constructs a Random Symmetric Multisegment for values 1-7.
length=7
M=[]
for x in range(0,4):
    l=np.random.randint(1,7)
    start=np.random.randint(0,length-l)
    if length +1 == 2*(start+1)+(l-1):
        S=[]
        for z in range(0,l):
            S.append(start+1+z)
        M.append(S)
    else:
        S1=[]
        S2=[]
        for z in range(0,l):
            S1.append(start+1+z)
            S2.append(length-start-z)
        M.append(S1)
        M.append(sorted(S2))
M1=list(M)
M3=list(M)
P=MStoMT(M1)
Q=MStoMT(M3)
Dual1=MW(P)
Num1=len(Dual1)
RT1=MTtoRT(Q)
DMT1=MStoMT(Dual1)
DRT1=MTtoRT(DMT1)
# Code 10 goes here
```

In order to the verify the theorems, we check that the output file remains empty. If a counterexample exists then it will be printed as follows in the output file:

```
Counter:
RT1:[[3, 3, 5, 4, 5, 3, 3], [2, 3, 4, 4, 3, 2], [2, 2, 4, 2, 2], [1, 2, 2, 1],
                                        [1, 0, 1], [0, 0], [0]]
RT2:[[3, 3, 5, 4, 5, 3, 3], [2, 3, 4, 4, 3, 2], [2, 2, 4, 3, 2], [1, 2, 2, 2],
                                        [1, 0, 1], [0, 0], [0]]
DRT1:[[3, 3, 5, 4, 5, 3, 3], [1, 3, 3, 3, 3, 1], [1, 1, 2, 1, 1], [0, 1, 1, 0],
                                         [0, 0, 0], [0, 0], [0]]
DRT2:[[3, 3, 5, 4, 5, 3, 3], [1, 3, 3, 3, 3, 1], [1, 1, 2, 1, 1], [0, 1, 1, 0],
                                         [0, 0, 0], [0, 0], [0]]
```

The following code is a slightly modified version of the Code 11a, which instead takes a given top row of the triangle and separately outputs the rank triangles for α which satisfy and break the boundary relation. This code is in support of the results found in **Section 4.3**.

```
# Code 12: Categorising the Partial Ordering Relation of Rank Triangles for a
                                        given Top Row.
Output = open("OutputSatisfied.txt", "w")
Output2= open("OutputCounter.txt", "w")
Top=[3,5,5,3]
AllAd=all_admissible(Top)
Output.write('Top Line:'+str(Top)+'\n')
Output.flush()
Output2.write('Top Line:'+str(Top)+'\n')
Output2.flush()
for t in AllAd:
    T=list(t)
    t1=list(t)
    t2=list(t)
    BR=True
    p=RTtoMT(t1)
    q=RTtoMT(t2)
    Dual1=MW(p)
    RT1=MTtoRT(q)
    DMT1=MStoMT(Dual1)
    DRT1=MTtoRT(DMT1)
    AllDRT2s=all_multisegments2(RT1)
    for pi in AllDRT2s:
        RT2=list(pi)
        P2=RTtoMT(RT2)
        RT2a=list(pi)
        Dual2=MW(P2)
        Num2=len(Dual2)
        DMT2=MStoMT(Dual2)
        DRT2=MTtoRT(DMT2)
        if DualAlesseqDualB(DRT1, DRT2)==True:
            if RT1 != RT2:
                BR=False
    if BR==True:
        Output.write(str(T)+str(DRT1)+'\n')
        Output.flush()
    elif BR==False:
        Output2.write(str(T)+str(DRT1)+'\n')
        Output2.flush()
```

In **Section 4.3.2**, we seek to classify all maximal endoscopic decompositions for any dimension which contradict the partial ordering relation. To implement this we can construct all possible multisegments for a given number of segments contained in it and check whether they are an endoscopic decomposition of a previous result using the following functions. Note this code already implements **Proposition 4.3.7** to improve computational efficiency, however the codes have also been implemented without these conditions.

```
# Function 13: Constructs all Multisegments for a given number of Segments

# dim= Total number of elements in rank triangle; numseg= Number of segments;
                                        Dim = Dimension of Space
def Construction(dim,numseg,Dim):
    Perms=[]
    Nim=min(Dim-2,numseg)
    for n0 in range(0,Nim+1):
        Perms.append([n0])
    for m in range(1, dim-1):
        Perms1=[]
        for n1 in Perms:
            Nim=min(Dim-2,numseg-np.sum(n1))
            for n2 in range(0,Nim+1):
                N1=n1+[n2]
                Perms1.append(N1)
        Perms=deepcopy(Perms1)
    Perms1=[]
    for n1 in Perms:
        if numseg-np.sum(n1)==0:
            N1=n1+[0]
            Perms1.append(N1)
    Perms=deepcopy(Perms1)
    return Perms
```

```
# Function 14: Takes a list and creates a Triangle Array
def gen_list_of_lists(original_list, new_structure):
    assert len(original_list) == sum(new_structure), \
        "The number of elements in the original list and desired structure don't
                                                match"
    list_of_lists = [[original_list[i + sum(new_structure[:j])] for i in range(
                                        new_structure[j])] \
                     for j in range(len(new_structure))]
    return list_of_lists
```

```
# Function 15: Checks whether one Multisegment will form an Endoscopic
                                        Decomposition of another
def Check(MT,prim):
    check=True
    for lis in range(0,len(prim)):
        for el in range(0, len(prim[lis])):
            W=MT[lis][el]- prim[lis][el]
            if W <0:
                check=False
    return check
```

```python
# Function 16: Checks whether an Endoscopic Decomposition [A = (A-B) union B] is
                                        such that [dual(A) = dual(A-B) union
                                       dual(B)]
from copy import deepcopy
def Decomp(TM,prim):
    check=True
    MT1=list(TM)
    prim1=list(prim)
    W= deepcopy(MT1)
    for lis in range(0,len(prim)):
        for el in range(0, len(prim[lis])):
            W[lis][el]=MT1[lis][el]- prim[lis][el]
    DualPrim=MW(deepcopy(prim1))
    DualMT=MW(deepcopy(MT1))
    DualW=MW(W)
    MTDualPrim=MStoMT(DualPrim)
    MTDualMT=MStoMT(DualMT)
    MTDualW=MStoMT(DualW)
    for lis in range(0,len(MTDualPrim)):
        for el in range(0, len(MTDualPrim[lis])):
            if MTDualMT[lis][el] != MTDualPrim[lis][el] + MTDualW[lis][el]:
                check=False
    return check
```

We can then use the following code (currently set for quiver representations of length 5) in order to obtain the results in **Table 4.1** and establish the corollaries found in **Section 4.3.2**.

```
# Code 17: Number of Indecomposable Counter Examples
RTForm = [5,4,3,2,1]
Output = open("Output5.txt", "w")
m=15
Primitives=[]
for N in range(3,20):
    n=int(N)
    Answer=Construction(15,n,5)
    Possible=[]
    for q1 in Answer:
        if q1.count(0)<13:
            q1=list(q1)
            Q1 = gen_list_of_lists(original_list=q1, new_structure=RTForm)
            Possible.append(Q1)
    for tm in Possible:
        t=MTtoRT(tm)
        T=list(t)
        t1=list(t)
        t2=list(t)
        BR=True
        p=RTtoMT(t1)
        q=RTtoMT(t2)
        Dual1=MW(p)
        RT1=MTtoRT(q)
        DMT1=MStoMT(Dual1)
        DRT1=MTtoRT(DMT1)
        AllDRT2s=all_multisegments2(RT1)
        for pi in AllDRT2s: #Checks for counter example
            if BR==True:
                RT2=list(pi)
                P2=RTtoMT(RT2)
                RT2a=list(pi)
                Dual2=MW(P2)
                Num2=len(Dual2)
                DMT2=MStoMT(Dual2)
                DRT2=MTtoRT(DMT2)
                if DualAlesseqDualB(DRT1, DRT2)==True:
                    if RT1 != RT2:
                        BR=False
        if BR==False:  #Check for previous examples contained in
            if len(Primitives)!=0:
                Checks=True
                for x in Primitives:
                    TFChecks=Check(tm, x)
                    if Checks==True:
                        if TFChecks == True:
                            TFEndo=Decomp(tm,x)
                            if TFEndo==True:
                                Checks=False
                if Checks==True:
                    Primitives.append(tm)
            else:
                Primitives.append(tm)
    Output.write(str(n)+":"+str(len(Primitives))+ '\n' + str(Primitives) +'\n')
    Output.flush()
```